PATENT ABSTRACTS OF JAPAN

(11)Publication number : 05-108319
(43)Date of publication of application : 30.04.1993

(51)Int.Cl. G06F 9/06

(21)Application number : 03-270713
(22)Date of filing : 18.10.1991

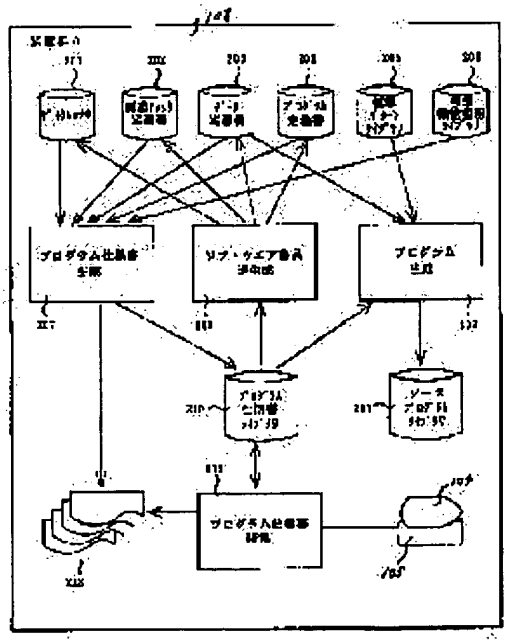
(71)Applicant : HITACHI LTD
(72)Inventor : MORIOKA YOSUKE
ONO OSAMU
NISHIOKA CHIHARU
HASHIMOTO NAOKI
NAKAGAWA TAMOTSU

(54) AUTOMATIC PROGRAM GENERATING SYSTEM USING DATA-BASED SOFTWARE PARTS

(57)Abstract:

PURPOSE: To improve the productivity and the reliability of software by generating automatically a program and its specifications with the use of the data item-based software parts.

CONSTITUTION: A function 207 selects automatically the software parts defined in a dictionary 201 and a relative check definition form 202 and produces the program specifications. A function 209 produces a source program by synthesizing the program skelton parts stored in a standard pattern library 205 with the software parts based on the specifications produced by the function 207. A function 212 edits the program specifications. Then a function 208 changes the software parts based on the program specifications. In such constitution, the reutilization ratio and the productivity of the software are improved. Furthermore the matching properties are automatically secured among the software parts, the specifications produced from the software parts, and the source program for improvement of the reliability of these items.



LEGAL STATUS

[Date of request for examination]	25.08.1998
[Date of sending the examiner's decision of rejection]	
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]	
[Date of final disposal for application]	
[Patent number]	3186117
[Date of registration]	11.05.2001
[Number of appeal against examiner's decision of rejection]	
[Date of requesting appeal against examiner's decision of rejection]	
[Date of extinction of right]	

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

CLAIMS

[Claim(s)]
[Claim 1] A means to store in the file which divides independent processing of as opposed to each data item for software into some classes, defines as software components, and is called a dictionary to a data item unit, A means to store in the file which defines related check processing in which the validity of data is judged based on the correlation between two or more data items, as software components, and is called a related check definition document, A means to manage the software components stored in these dictionaries and a related check definition document based on a data item name, A means to store in the file which defines information, such as a file, and the attribute of a document, a layout, and is called a data definition document, A means to store in the file which defines the information on input/output media as a program unit, and is called a program definition document, Pay one's attention to the special feature of a control structure, and processing programs, such as a screen, a document, and a file, are divided into the class of shoes. A means to store in the file which components-izes as a program skeleton standard pattern which does not have operation individual processing in itself, and is called a standard-pattern library, A means to store the standard function recital article which met program skeleton standard-pattern components in the file called a standard function explanation library, A dictionary, a related check definition document, a data definition document, a program definition document, And a means to generate automatically the program specification which chooses software components from a standard function explanation library based on the data item name in the input/output media of the user program made into an object, and consists of some kinds of specification, Based on program specification, program skeleton standard-pattern components, The program automatic generation method by the data core mold software components characterized by having a means to generate a source program automatically by choosing and compounding the software components in a dictionary, and software components related check definition in the letter.
[Claim 2] The program automatic generation method by the data core mold software components characterized by having the means which divides the independent processing to the data item in a dictionary into three kinds, a check, input edit, and output edit, in the program automatic generation method by the data core mold software components of claim 1.
[Claim 3] The program automatic generation method by the data core mold software components characterized by having the means which enables correction of modification and the addition to the generated program specification, deletion, etc. in the program automatic generation method by the data core mold software components of claims 1 and 2, and a means to generate a source program automatically based on the program specification modified here.
[Claim 4] The program automatic-generation method by the data core mold software components characterized by having a means to correct the content of a definition of the software components automatically stored in the dictionary, the related check definition document, the data definition document, and the program definition document from the corrected program specification in the program automatic generation method by the data core mold software components of claims 1, 2, and 3.
[Claim 5] The program automatic-generation method by the data core mold software components

characterized by to have the means which makes it possible to correct program specification using general-purpose software on the equipment different from the equipment which generated program specification to the generated program specification in the program automatic generation method by the data core mold software components of claims 1, 2, 3, and 4.

[Translation done.]

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Industrial Application] This invention relates to the suitable software-development exchange approach to increase the efficiency of a new software development by starting the productivity drive of software, especially components-izing the existing software, and generating program specification automatically from these components, and generating a program automatically from program specification.
[0002]
[Description of the Prior Art] There is a method of reusing the software of existing [one] of the leading approach for improving software productivity, and developing new software. When reusing this existing software and developing new software, it is common to reuse in the phase of operation system creation or the phase of a programming, and to correct by analyzing the document and source program with which the programmer described the specification of the existing software.
[0003] However, also in the software of operation of the same kind, in order to add the correction accompanying a difference of two or more users' requirement specification to one software with it difficult [for the requirement specification to be various and to pinpoint a correction part only from a document or a source program by the approach of the above reuse] by the user, there was a fault of software maintainability deteriorating.
[0004] For such a fault, reuse of the existing software was given up, software was completely newly developed even in the software development of operation of the same kind, in many cases, and the productivity drive was barred.
[0005] By subdividing in that function as one approach of solving this problem paying attention to software, it components-izes and there is a method of generating a new program automatically by defining matching with these components and programs, respectively. When dividing software into a small-scale components group by this approach and developing new software, it becomes possible to reuse the existing software on components level, and could respond by choice of components, modification, and addition to the difference in the application of software, and the difference in requirement specification.
[0006] In addition, the pages 19-24 "system development support-software "EAGLE"" of Hitachi criticism VOL.66 (March, 1984 issuance) etc. are one of those which are related as this kind of a technique.
[0007]
[Problem(s) to be Solved by the Invention] The following technical problems occur in the conventional approach.
[0008] The 1st technical problem in the conventional technique is in the succession approach to the means of components-izing of software itself, and the program specification document of software components design information.
[0009] Since fragmentation of a function was performing components-ization of software, in case components-ization was performed with the conventional technique of generating a program

automatically by components-izing the above-mentioned software and compounding it, it was difficult in what kind of viewpoint to subdivide a function or which level to use as one component, and to establish criteria clear about ********, and the division approach of software was various by the architect of components. For this reason, it was difficult to put a standardization of software components into practice.

[0010] Furthermore, since an architect or a programmer had to program 11 define matching with the target user program and components in order to generate a source program, a productivity drive and improvement in dependability were not fully able to be aimed at.

[0011] It was that by which most creation of a program specification document, on the other hand, depends creation of these software components on human being's independent handicraft although some means to generate them to the means for creating software components and a source program were developed. Although there is the approach of managing as an electronic filing document which created the program specification document using computers only for document preparation, such as the so-called word processor, and fair copy and storage of a document can be supported as for this, the program-design activity itself is still based on a program-design person's handicraft including the electronic-filing-document creation activity. It is impossible to realize automatic succession of the design information from the design process of software components to a program-design process by such creation approach of a program specification document. For this reason, since check that it is difficult and the source program and program specification document which are further generated from software components and it are in agreement is also left to the handicraft of a programmer, the productivity drive of a program specification document creation activity also serves as hindrance of the improvement in software reliability.

[0012] The 2nd technical problem in the conventional technique is not to have a means to generate the source program as a program specification document, when a different specification from the software components beforehand prepared for the program specification document is described.

[0013] Even if it described a different specification from the software components beforehand prepared for the program specification document for some reasons of doing most program specification document creation activities by handicraft in the conventional technique as the 1st technical problem described, and the program for generation including exception handling for this reason, it was impossible to have made this reflect in source program generation. Therefore, since it was necessary to correct the source program with which the programmer was generated according to a program specification document in such a case, a productivity drive and improvement in dependability were not fully able to be aimed at.

[0014] When modification arises in the specification of an object user program, the 3rd technical problem in the conventional technique is that software components are automatically uncorrectable according to the specification described by the program specification document, even if it corrects a program specification document. Moreover, it is also the same as when the software components itself are the right specifications of original [specification / which the specification has mistaken by the poor definition and was described by the program specification document] as another case. Therefore, since it was necessary to correct the source program with which the programmer was generated according to a program specification document in such a case, and to also make correction of software components further, it became the hindrance of the improvement in dependability of software components, as a result had also become the hindrance of software components reuse.

[0015] The 4th technical problem in the conventional technique is in the employment approach and operability of the software which supports development by components-izing of such software.

[0016] if those who can perform it only by the actuation for using the software (such software -- a following software development tool -- or it only being called a tool) which supports a software development which was described until now using the equipment with which the tool concerned was incorporated, and use the tool did not follow the operating instructions which the tool surely defined, when there were, they did not become. [no] For this reason, it is necessary to learn the operating instructions of the proper which a software-development person's activity was restricted to an installation, an employment time zone, etc. of equipment when the tool was incorporated on the other

hand by a manager having to perform plant-and-equipment investment corresponding to a software-development person's manpower, and the tool defined, as a result has become the hindrance of a software productivity drive.
[0017] By establishing the fixed criteria for which it does not depend on an architect's individuality or the special feature of application in components-izing of software, and establishing the approach of managing components in the format corresponding to these criteria further, the 1st object of this invention standardizes the components of software, and is to make reuse of the existing software easy.
[0018] Furthermore, it is in inheriting the design information of software components automatically to program specification and a source program, and improving the productivity and dependability of software by making it possible to choose software components required for the program concerned based on the name of a data item, to generate program specification automatically, and to generate a source program automatically from this program specification, without an architect or a programmer defining matching with a user program and software components.
[0019] The 2nd object of this invention is by enabling correction of modification, an addition, deletion, etc. to the generated program specification, and making it possible to generate the source program as program specification automatically, also when the specification indicated by the program specification after correction differs from the specification of the software components prepared beforehand to improve the productivity and dependability of software.
[0020] The 3rd object of this invention is by keeping the consistency of program specification and software components automatic, and improving the dependability of software components, as a result making reuse easy by enabling correction of program specification as mentioned above, and making it possible to correct software components automatically according to the specification indicated by the program specification after correction if needed to also raise the productivity of software.
[0021] Modification to the program specification which stated the 4th object of this invention for the 2nd object, Transfer methods, such as an addition and deletion, by so-called personal computers and workstations other than the specific equipment with which the software development tool concerned was incorporated By making it possible to carry out using the software for the so-called spreadsheets of a general purpose, or the software for document preparation Without being restricted to the employment conditions of the equipment of specification [a software-development person], while aiming at reduction of the plant-and-equipment investment for using the software development tool concerned And it is in making it possible to offer the work environment which can perform a software development using the general-purpose software which already got used and was familiar, even if it does not learn the operating instructions of a tool proper, as a result raising the productivity of software.
[0022]
[Means for Solving the Problem] In order to attain the 1st object, by this invention, divide independent processing of as opposed to each data item for software into some classes, and it is defined as a data item unit as software components. Related check processing in which the validity of data is judged based on the correlation between two or more data items to be a means to store in the file called a dictionary is defined as software components. It is characterized by having a means to store in the file called a related check definition document, and a means to manage the software components stored in these dictionaries and a related check definition document based on a data item name.
[0023] The processing hereafter defined in the software components based on data items, and calls and these components in accordance with the software components stored in the above-mentioned dictionary and a related check definition document is called processing based on data items.
[0024] Furthermore, a means to store in the file which defines information other than a means to define the software components based on [above-mentioned] data items, such as a file, and the attribute of a document, a layout, and is called a data definition document, A means to store in the file which defines the information on input/output media as a program unit, and is called a program definition document, Pay one's attention to the special feature of a control structure, and processing programs, such as a screen, a document, and a file, are divided into the class of shoes. A means to store in the file which components-izes as a program skeleton standard pattern which does not have operation individual

processing in itself, and is called a standard-pattern library, It is characterized by having a means to store the standard function recital article which met program skeleton standard-pattern components in the file called a standard function explanation library.
[0025] Furthermore, the above-mentioned dictionary, a related check definition document, a data definition document, Software components are chosen from a program definition document and a standard function explanation library based on the data item in the input/output media of the user program made into an object. A means to generate automatically the program specification which consists of some kinds of specification, It is characterized by having a means to generate a source program automatically by choosing and compounding program skeleton standard-pattern components, and the software components in a dictionary and software components related check definition in the letter based on program specification.
[0026] As a means for attaining the 1st object under still stricter standardization criteria, it is characterized by having the means which divides the independent processing to the data item in a dictionary into three kinds, a check, input edit, and output edit, by this invention.
[0027] In order to attain the 2nd object, in this invention, it is characterized by having the means which enables correction of modification and the addition to the generated program specification, deletion, etc., and a means to generate a source program automatically based on the program specification modified here.
[0028] In order to attain the 3rd object, in this invention, it is characterized by having a means to correct the content of a definition of the software components automatically stored in the dictionary, the related check definition document, the data definition document, and the program definition document from the corrected program specification.
[0029] In order to attain the 4th object, in this invention, it is characterized by having ****** which makes it possible to correct program specification to the generated program specification on the equipment different from the equipment which generated program specification using general-purpose software.
[0030]
[Function] In order to attain the 1st object, we defined related check processing in which divided independent processing of as opposed to each data item for software into some classes, and defined it as a data item unit as software components by this invention, and the validity of data was judged based on the correlation between two or more data items, as software components, and decided to manage software ****** based on these data based on a data item name. On the other hand, about the software components which are not contained in the software components based on these data The software components which defined the file, an attribute, a layout of a document, etc., The software components, screen which defined the input/output-media information on a program unit, Pay one's attention to the special feature of a control structure, and processing programs, such as a document and a file, are divided into the class of shoes. We decided to divide and define operation individual processing as the software components called the program skeleton standard pattern which it does not have in itself, and the software components called the standard function recital article which met program skeleton standard-pattern components.
[0031] The clear fixed criteria of dividing software focusing on a data item in components-izing of software by the above about the division approach of the software for [the conventional technique / with an architect's individuality or the special feature of application] various components-ization can be established. Since the approach of managing in the format which agreed on these criteria by furthermore storing these software components in the file which had specific structure, respectively is establishable, the components of software can be standardized, and it can become possible to make reuse of the existing software easy, as a result the productivity drive of software can be planned, and the technical problem of the conventional technique can be solved.
[0032] Furthermore, the above-mentioned dictionary, a related check definition document, a data definition document, Software components are chosen from a program definition document and a standard function explanation library based on the data item in the input/output media of the user

program made into an object. The program specification which consists of some kinds of specification is generated automatically. Based on program specification Program skeleton standard-pattern components, It considered as the method which generates a source program automatically by choosing and compounding the software components in a dictionary, and software components related check definition in the letter.

[0033] Therefore, in a Prior art, it becomes possible to be able to generate a source program automatically, to automate the program specification creation activity for which most depended on an architect's handicraft in the Prior art, and to realize automatic succession of the design information from the design process of software components to a program-design process, without an architect or a programmer performing matching with the program and software components which had to define one 1 user program. By this, the productivity and dependability of software can be improved and the technical problem of the conventional technique can be solved.

[0034] As a means for attaining the 1st object under still stricter standardization criteria, it decided to divide the independent processing to the data item in a dictionary into three kinds, a check, input edit, and output edit, by this invention. It becomes possible to limit further the structure of the software components stored in a dictionary by this, and the productivity and dependability of software components improve, therefore reuse also becomes easy.

[0035] Since the 2nd object is attained, in this invention, correction of modification and the addition to the generated program specification, deletion, etc. is enabled, and a source program can be generated automatically based on the program specification modified here.

[0036] For some reasons of the program for generation including exception handling by this Since it becomes possible to make this reflect in source program generation also when a different specification from the software components beforehand prepared for program specification is described, It becomes unnecessary to correct the source program generated like the conventional technique as software components according to a program specification document, the productivity drive of software and improvement in dependability can be aimed at, and the technical problem of the conventional technique can be solved.

[0037] In order to attain the 3rd object, in this invention, it made it possible to correct the content of a definition of the software components automatically stored in the dictionary, the related check definition document, the data definition document, and the program definition document from the corrected program specification. Like the case stated with the 3rd technical problem in the conventional technique, for the reason of modification occurring in the specification of an object user program, also when program specification is corrected, in accordance with the specification described with program specification, it can correct to software components automatically. For this reason, in such a case, with the conventional technique, the correction of software components which was being done manually can be automated, the productivity of software also improves with the improvement in dependability of software components, and the technical problem of the conventional technique can be solved.

[0038] In order to attain the 4th object, in this invention, it made it possible to correct program specification to the generated program specification on the equipment different from the equipment which generated program specification using general-purpose software. It becomes possible to perform transfer methods, such as modification and the addition to program specification, and deletion, using the software for the so-called spreadsheets of a general purpose, or the software for document preparation by so-called personal computers and workstations other than the specific equipment with which the software development tool concerned was incorporated by this. Therefore, while aiming at reduction of the plant-and-equipment investment for using the software development tool concerned, without being restricted to the employment conditions of specific equipment to a software-development person, and even if it do not learn the operating instructions of a tool proper, it can become possible to offer the work environment which can perform a software development using the general-purpose software which already got used and be familiar, as a result the productivity of software can be raised, and the technical problem of the conventional technique can be solved.

[0039]

[Example] Hereafter, the example of this invention is explained to a detail using drawing.
[0040] First, the means for realizing the hardware configuration for realizing this example using drawing 1 and drawing 2 and some functions which this example has is explained.
[0041] Drawing 1 shows the hardware configuration of the system for realizing this invention. The equipment group A (108) is an equipment group for updating the software components which are generation of program specification or a source program, and the generator information on the program specification from program specification that the generated program specification was corrected and revised (reverse generation being called in this example). Moreover, the equipment group B (118) is used when making correction of program specification on a different equipment group from the equipment group A. The optimal thing is chosen from the environment where the user was placed although it is desirable as an equipment group B to use the so-called personal computer and the so-called workstation. moreover, it is also possible to see two or more sets of equipment groups B, and to use them to the equipment group A, -- carrying out -- reverse -- the equipment group B -- it is nothing and it is also possible to perform function concerning this invention and actuation using the equipment group A altogether.
[0042] First, the equipment group A (108) is explained along with drawing 1 . The equipment group A is equipped with CPU (103) as the central equipment. The input unit for performing the data input at the time of editing the program specification which performed the input for ordering it activation of generation, or was generated (105), The display for displaying the data of the generated program specification etc. (104), and in connecting the printer (102) for outputting the generated program specification to CPU as the input/output equipment and also using the equipment group B The communication device (107) for transmitting the generated program specification to the equipment group B, or receiving the program specification edited by the equipment group B is connected to CPU.
[0043] While performing to CPU external storage (106) for storing the software components which serve as the generator again at a program specification generate time, and the program specification and the source program after generation, and generation and edit of program specification, the memory (101) for storing a program and data is connected. The file of the dictionary (201) shown in drawing 2 , a related check definition document (202), a data definition document (203), a program definition document (204), a standard-pattern library (205), a standard function explanation library (206), a program specification library (210), and a source program library (211) is included in the data stored in external storage (106). It is the same as that of the equipment group A also about each equipment of the equipment group B (118). However, the data stored in external storage (116) can be made only into a program specification library (210).
[0044] This example is realized by four functions, the program specification generation function (207) shown in drawing 2 , a program generation function (209), a program specification edit function (212), and a software components reverse generation function (208).
[0045] A program specification generation function (207) is stored in the program specification library (210) which is similarly on external storage (106), or outputs the program specification (213) which inputted the dictionary (201) on external storage (106), the related check definition document (202), the data definition document (203), the program definition document (204), and the standard function explanation library (206), generated program specification, and was generated to a printer (102).
[0046] A program generation function (209) inputs the program specification library (210), data definition document (203), and standard-pattern library (205) on external storage (106), generates a source program, and stores it in the source program library (211) on external storage (106).
[0047] A program specification edit function (212) is an editor which inputs the program specification library (210) on external storage (106), and corrects addition of program specification, modification, deletion, etc. using an input unit (105) and a display (104).
[0048] A program specification edit function (212) can also be used on the equipment group B (118). In this case, the program specification library (210) of the equipment group A (108) is transmitted to the external storage (116) of the equipment group B from the external storage (106) of the equipment group A, using the display (114) and input unit (115) of the equipment group B, the program specification

library (210) on the external storage (116) of the equipment group B is edited, and the corrected program specification is outputted to the printer (112) on the equipment group B.
[0049] The program specification library (210) furthermore edited by the equipment group B can be again transmitted on the external storage (106) of the equipment group A, and the software components reverse generation function (208) stated to a program generation function (209) or a degree by the equipment group A can be used.
[0050] It is also realizable to use the editor of dedication, in order to realize a program specification edit function (212) in more than, and also to carry out using the software for document preparation, the existing general-purpose software, for example, so-called software for spreadsheets, with the conventional technique, and let it be the range of this invention also in this case.
[0051] A software components reverse generation function (208) inputs the program specification library (210) on external storage (106), and updates the software components stored in the dictionary (201) which is similarly on external storage (106), the related check definition document (202), the data definition document (203), and the program definition document (204) if needed.
[0052] In this example, program specification (213) consists of five kinds of specification, a program manipulation schematic diagram (drawing 18), a program function explanatory view (drawing 19), a check condition table (drawing 20), a related check condition table (drawing 21), and an edit condition table (drawing 22).
[0053] In addition, although this example explains five kinds as a case above, besides this, the document relevant to these programs, such as file specification, record specification, document specification, screen specification, database specification, subroutine connection specification, and a resource on other equipments, access specification to a process, can be outputted by the Prior art, and it considers as the range of this invention also in this case.
[0054] In advance of explanation of the concrete procedure of the program specification generation by this invention, the definition-part article stored in a dictionary (201), a related check definition document (202), a data definition document (203), a program definition document (204), a standard-pattern library (205), a standard function explanation library (206), and each library is explained to a detail.
[0055] In addition, drawing 6 is the schematic diagram of the example of a user program made applicable [of this invention] to application. A program 602 reads the input file "KYFIL" (601) which has DS (605), checks validity of data, and normal data are edited into the output document "KYLST" (603) which has a format of a layout 606, respectively, and it outputs them to it at the output document "KYERR" (604) with which unjust data have a format of a layout 607. Henceforth, this example is explained for this example program.
[0056] A dictionary (201) is explained using drawing 7 - drawing 11 . With a dictionary, the software components of the independent processing about each data item are defined and stored in a data item unit. Here, a data item points out the data with professional semantics, such as a name and a date of birth, of a smallest unit. Independent processing is actuation performed only to this data item, and is processing which does not participate in any data items other than itself.
[0057] There are a data item definition part (701), a check processing definition part (706), an input edit definition part (707), and an output edit definition part (708) in the software components of a dictionary. Here, check processing, input edit processing, and output edit processing are defined as follows. That is, the processing which checks the format of data and the validity of data to the input from an input document or an input screen is check processing, the processing changed to the format of storing the inputted data in a database or a file is input edit processing, and the processing change to the format of outputting a database and the data on a file to an output document or an output screen is output edit processing.
[0058] This example shows the example of a definition over the data item of "SIMEI-NO" (drawing 7), "SIMEI" (drawing 8), "NYUSYA-YMD" (drawing 9), "SYOKUI-CD" (drawing 10), and "SIKYU-GK" (drawing 11).
[0059] The attribute which consists of a data item name, a Japanese name, a format, and a digit count, and the comment which specifies the application and semantics of a data item are defined as a data item

definition part (701). The data item name was "SIMEI-NO", a Japanese name of (703) is a "name number", (702) defined that the attribute of (704) of data was 9 figures of a figure by the example of drawing 7 , and (705) defines the semantics of a data item by it.
[0060] It is defined as a check processing definition part (706) by the error code and error message which are used in case it tells to a user that unjust things are the conditions for judging that data are unjust. Similarly, it is the keyword which shows that (709) checks by calling a subroutine "SUB01" to the input about this data item, and an error code when (710) is an error is "ER10", and (711) defines that an error message is "not right [a name number]" by the example of drawing 7 .
[0061] An input edit definition part (707) is explained using drawing 8 . This example defines performing input edit about this data item by calling a subroutine "SUB02" by describing it as "SUB (SUB02), SIMEI-NO" to (707). Similarly, about the output edit definition part, by describing it as "DATE" to (708), the example of drawing 9 defines changing and outputting an output form to "YY.MM.DD" (YY: a year, MM:moon, DD:day), in case output edit about this data item is performed. As mentioned above, for the definition of the edit approach of check conditions, and input edit and output edit, it describes using keywords, such as "SUB" and "DATE." In the case of check conditions, in the case of the edit approaches, such as a subroutine call, and a numeric check, an alphabetic-character check, a range check, keywords are a character string showing some common software components, such as a subroutine call, and a formula, character string connection edit, and its syntax rule.
[0062] Next, the related check definition-part article stored in a related check definition document (202) using drawing 12 is explained. A related check definition-part article defines the software components of the related check processing between two or more data items by the format of a decision table.
[0063] This example is an example of the related check definition-part article about a data item "SIKYU-GK" and "SYOKUI-CD." (801) defines that the target data item name is "SIKYU-GK", and (802) defines that a related data item name is "SYOKUI-CD." In addition, of course, there may be two or more related data item names. (803) is the condition definition section of a related check, and (807) is the error-processing definition part of a related check.
[0064] A data item name (804), conditions (805), and a related matrix (806) are defined as the condition definition section (803). An error code (808), an error message (809), and a related matrix (810) are defined as an error-processing definition part (807).
[0065] In addition, conditions (805) are described using a keyword like the software components of a dictionary. this example defines what "SIKYU-GK does not come out size from 80000, it sets an error code to ER50 from 'A03' when SYOKUI-CD is size, and an error message is 'an allowance is below criteria'", and ", as for an error code, it setting to ER60 from 'A03', when SYOKUI-CD is not size, SIKYU-GK being size from 80000 and an error message being 'the allowance is over criteria'."
[0066] Next, a data definition document (203) is explained. A data definition document stores the document definition-part article, define the file components, and record definition components which defined the information on a file or a document. Hereafter, such definition-part articles are explained.
[0067] A document definition-part article is explained using drawing 13 (a). This example defines the information about the document of the "salary list" (606) in drawing 6 . There are a document attribute definition part (901) and a document layout definition part (907) in a document definition-part article. A document definition-part name of article is "KYLST", (902) defined that a document name of (903) was a "salary list", (905) defines the record length and (906) defines the number of the print lines of the detail business per page. The repeat factor (910) of the initial line (908) when setting forth a data item, an initiation train (909), and a detail line, a data item name (911), and an attribute (912) are defined on a document at a document layout definition part (907).
[0068] Although drawing 13 (b) is similarly a document definition-part article about the "error list" (607) in drawing 6 , since the content is the same as that of drawing 13 (a), explanation is omitted.
[0069] Explanation of define the file components and record definition components is given using drawing 14 .
[0070] This example defines the file of (601) in the example program of drawing 6 , and its record format (605). As for (1002), a define the file components name is "KYFIL", and a file name of (1003) is

a "salary file." The file symbolic name which uses (1004) within a program is "KYUYO-FILE." (1005) defined that the record definition components name which defined the record format to this file was "KYREC", and defines the record length (1006), the block length (1007), record format (1008), etc. as an attribute of a file.
[0071] There are a record attribute definition part (1009) and a record layout definition part (1012) in record definition components, and it defines that as for (1010) a record definition components name is "KYREC" and a record name of (1011) is a "salary record" as them. The level number (1013), a data item name (1014), and an attribute (1015) are defined as a record layout definition part.
[0072] In addition, although the program of batch processing is made into the example and the three above-mentioned kinds are used as a data definition document in this example In the system which uses the screen and database other than a file or a document A screen definition-part article, In the case of the system corresponding to a client server model, the resource on other equipments, the access definition-part article to a process, etc. can define a database definition-part article similarly again, and it is contained in this invention also about this.
[0073] A standard-pattern library (205) is explained using drawing 16 .
[0074] Its attention is paid to a control structure, processing programs, such as a screen, a document, and a file, are divided into the class of shoes, and the program skeleton standard-pattern components (it is hereafter called a standard pattern) components-ized as a pattern which does not include operation individual processing in itself are stored. This example is the standard pattern of the type which reads a file, checks and carries out another **** output at normal data and error data. (1201) is a program skeleton part article whose (1202) are a standard-pattern name and is a body of a standard pattern. The check processing section (1203), the related check processing section (1204), and the edit processing section (1205) are processings according to operation individual, and are a part into which these are not included in a standard pattern but a program generation function (209) develops the software components based on data items.
[0075] A standard function explanation library (206) is explained using drawing 17 . With a standard function explanation library, the standard function recital article which met the standard pattern is stored. This example is a standard function recital article corresponding to the standard pattern "CHK01" explained above. (1301) is a standard function recital name of article, and (1302) is description of the processing part of a functional description. In addition, the standard function recital article and the standard pattern have taken the 1 to 1 response, therefore a standard function recital name of article (1301) is a corresponding standard-pattern name and a corresponding homonym.
[0076] The program definition-part article stored in a program definition document (204) is explained using drawing 15 . This example defines the program specification of (602) in the example program of drawing 6 . (1101) is a program attribute definition part and (1106) is an input/output-media definition part. A program symbolic name is "PROG01", a program (1103) Japanese name is "salary list creation", and (1102) defines that the standard-pattern name which uses (1104) by the program is "CHK01." The comment which specifies a program manipulation outline is defined as (1105).
[0077] when using the define the file components name and document definition-part name of article corresponding to input/output media for an input/output-media definition part (1106) (1107) within each I/O partition (it is hereafter called an I/O partition) (1108), an activity partition (1109), a logical unit name (1110), and a program, it adds to a data item -- a prefix (1111) definition is carried out.
[0078] Here, an activity partition (1109) is a partition which shows that it is the file used for a certain special application in each standard pattern. Although the activity partition of input/output media "KYERR" is defined as "ERR" in this example, this shows that it is the output destination change of an error code or an error message edited by check processing in the standard pattern "CHK01."
[0079] Hereafter, the procedure of program specification generation is explained to a detail using the flow chart of drawing 3 . The program definition-part article (shown in drawing 15) of the user program for the introduction generation (it abbreviates to this program below) is inputted (step 301), and the standard-pattern name (1104) currently used by this user program, an input/output-media definition-part name of article (1108), and an activity partition (1109) are acquired.

[0080] Next, a program manipulation schematic diagram is developed and outputted (302).
[0081] The expansion method of a program manipulation schematic diagram is explained using drawing 18 . A program name (1401), a symbolic name (1402), a standard-pattern name (1403), and a processing outline (1412) output what was acquired from the program definition-part article (drawing 15), respectively (1105 (1104 (1102 (1103)))).
[0082] The definition-part name of article (1406) of input/output media (1404), an I/O partition (1407), an activity partition (1408), a logical unit name (1409), and a prefix (1410) output what was acquired from the input/output-media definition part (1111 (1110 (1109 (1108 (1107))))) of a program definition-part article, respectively. The Japanese name (1405) of input/output media (1404) acquires and outputs a file name (1003) and a document name (903) based on the definition-part name of article (1107) of input/output media with reference to define the file components and a document definition-part article.
[0083] in this example, since the definition-part name of article (1107) is defined as "KYFIL" at the program definition-part article, it is alike also in this definition-part name of article, and with reference to define the file components (drawing 14 (a)), a "salary file" is acquired as a file name (1003), and it outputs to a Japanese name (1405). (The processing which acquires the Japanese name of a file or a document with reference to a definition-part article based on a definition-part name of article in this way is henceforth called Japanese Natori profit.) with reference to a dictionary (201), the Japanese name of a data item can be similarly acquired based on a data item name about a data item, and it is called Japanese Natori profit also about this -- based on the further above-mentioned information, it diagrams and the I/O relation of this program is outputted, as shown in a drawing (1411).
[0084] Next, a standard function recital article is inputted based on the standard-pattern name (1104) already acquired from the return program definition-part article to drawing 3 (303), and a program function explanatory view is developed and outputted (304).
[0085] The expansion method of a program function explanatory view is explained using drawing 19 . A program name (1501), a program symbolic name (1502), and a standard-pattern name (1503) output what was acquired from the program definition-part article (1104 (1102 (1103))) like the above, respectively. About an input-medium name (1504) and an output media name (1506) (1507) as well as the above, with reference to a definition-part name of article (1107), a Japanese name is acquired and a definition-part name of article and a Japanese name are outputted from the input/output-media definition part (1106) of a program definition-part article, respectively. Whether it is an input medium among input/output media or it is an output media call an output media hereafter what is an input medium and "O" about that whose I/O partition is "I" among input/output media that what is necessary is just to refer to a corresponding I/O partition (1108).
[0086] About description of processing (1505) of functional description drawing, that in which input/output media carried out Japanese Natori profit is compounded and outputted to the description section (1302) of processing of a standard function recital article.
[0087] Next, the generation judging of a check condition table and a related check condition table is performed based on the standard-pattern name already acquired from the return program definition-part article to drawing 3 (305). For example, the standard pattern "CHK01" currently used by this example is a pattern which reads an input file, checks the content and is written out to an output file. A standard pattern including such check processing of input data is judged with it being necessary to carry out the generation output of a check condition table and the related check condition table.
[0088] When judged with a check condition table and a related check condition table needing to be generated, a check condition table and a related check condition table are developed and outputted as follows.
[0089] First, the expansion approach of a check condition table is explained using drawing 20 .
[0090] The check condition table is developed about all the data items that were defined as the program definition-part article and that input define the file components and record definition components based on an input-medium definition-part name of article for every input medium, and are further used with the input medium, referring to a dictionary for every data item (306).
[0091] About the output method of a program name (1603) and a program symbolic name (1604), it is

the same as that of said program manipulation schematic diagram. This input-medium definition-part name of article acquired from the program definition-part article and the Japanese Natori profit thing based on it are outputted to the definition-part name of article (1602) and its Japanese name (1601) of an input medium. The information about the output media of the output destination change of the error code edited by check processing or an error message is outputted to the definition-part name of article (1606) and its Japanese name (1605) of an output media. For this reason, among the output medias defined as the program definition-part article, an activity partition (1109) judges what is "ERR" to be the output destination change of an error message, and outputs that output-media definition-part name of article and the Japanese Natori profit thing based on it.

[0092] From the software components (701) of a dictionary (201), an attribute (1608) is acquired in the column of an input item from the definition-part article of an input medium, and this data item name and its Japanese name (1607) are outputted to it.

[0093] It outputs to the column of a keyword (1609), check conditions (1610), an error code (1611), and an error message (1612) based on the content defined as the check processing definition part (706) of a dictionary (201) by the column of check processing. Since the edit approach is described by the software components of a dictionary by the keyword about the check condition column, the semantics which a keyword shows is changed and outputted to Japanese.

[0094] In this example, first, since the input-medium definition-part name of article (1107) of a program definition-part article is "KYFIL", with reference to corresponding define the file components (drawing 14 (a)), "KYREC" which is a corresponding record definition components name (1010) is acquired. Furthermore, corresponding record definition components (drawing 14 (b)) are inputted, the input item column and the check processing column are developed in order of the data item defined, and it outputs and goes.

[0095] In the record layout definition (1012) of "KYREC", since the data item name (1014) defined as the 1st is "SIMEI-NO", with reference to the corresponding software components (drawing 7) of a dictionary (201), the "name number" which is the Japanese name (703) of a data item is acquired first. Next, from the check processing definition part (706) of a dictionary, a keyword (709), an error code (710), and an error message (711) are acquired and outputted. "SUB (SUB01)" of a keyword (1609) is a keyword which shows the software components of a subroutine call, and SUB01 expresses the program symbolic name of the subroutine called. So, the semantics which a keyword indicates "Checks by calling a subroutine (SUB01)" is developed and outputted to Japanese at the check condition column (1610) (the processing which changes into Japanese the semantics which such a keyword shows is henceforth called keyword Japanese conversion).

[0096] A check condition table is generated by repeating the same procedure to the data item name (1014) of a record layout definition (1012) below. Next, a return related check condition table is generated to drawing 3 (307). About all the input media defined as the program definition-part article, define the file components and record definition components are inputted based on an input-medium definition-part name of article, and the generation judging of a related check condition table is performed for every data item about all the data items currently further used with the input medium.

[0097] Criteria generate the related check condition table to this related check definition-part article, only when all the associated data items (802) indicated by the related check definition-part article corresponding to that the related check definition-part article to this data item exists and this data item are included in the input medium used by this program. In addition, when there are two or more input media, the associated data item should just be included in one of the input media.

[0098] In this example, since "SYOKUI-CD" which a related check definition-part article exists about "SIKYU-GK" among the data item names (1014) in an input medium (drawing 12), and is indicated by the associated data subject name (802) of this related definition-part article is contained in the record definition components (drawing 14 (b)) to the input medium of this program, it judges with an input check condition table needing to be generated. Under the present circumstances, the input-medium definition-part name of article which becomes the input origin of an associated data item is acquired.

[0099] About data items other than "SIKYU-GK" in which a related check definition-part article does

not exist, it judges not generating a related check condition table.
[0100] When judged with a related check condition table needing to be generated, the related check condition table is developed referring to this related check definition-part article and the software components of a dictionary (201).
[0101] The expansion approach of a related check condition table is explained using drawing 21 . About the output method of a program name (1701) and a program symbolic name (1702), it is the same as that of said program manipulation schematic diagram. About the symbolic name (1706) and its Japanese name (1705) on an input medium, the data item name acquired from the data item name (804) of a related check definition-part article and the Japanese Natori profit thing based on it are outputted. About conditions (1707) and a matrix (1708), it acquires from the conditions (805) and matrix (806) of a related check definition-part article, and outputs. About an error code (1711), an error message (1712), and a matrix (1713), it acquires from the error code (808), error message (809), and matrix (810) of a related check definition-part article, and outputs.
[0102] About the definition-part name of article (1704) and its Japanese name (1703) of an input medium, the definition-part name of article of the input medium with which this data item is included already acquired at the time of a related check generation judging, and the thing which performed Japanese Natori profit to it are outputted.
[0103] About the definition-part name of article (1710) and its Japanese name (1709) of an output media, it outputs completely like the error code in a check condition table, and an error message output destination change.
[0104] Next, the expansion approach of an edit condition table is explained using drawing 22 .
[0105] all the combination of the output media and input medium which were defined as the program definition-part article -- receiving -- the -- combining -- ** -- it is alike and define the file components, record definition components, and a document definition-part article are inputted based on this output-media definition-part name of article, and the edit condition table is developed about all the data items currently further used by the output media, referring to the software components of a dictionary for every data item (step 308 of drawing 3).
[0106] About the output method of a program name (1803) and a program symbolic name (1804), it is the same as that of said program manipulation schematic diagram. This output-media definition-part name of article and this input-medium definition-part name of article which were acquired from the program definition-part article, and the Japanese Natori profit thing based on it are outputted to the definition-part name of article (1802) of an output media and its Japanese name (1801), the definition-part name of article (1806) of an input medium, and its Japanese name (1805), respectively.
[0107] About the column of an output item, an output item name (1807) outputs a data item name and the Japanese Natori profit thing based on it, and an attribute (1808) acquires and outputs the attribute (912) defined on the output media to this data item name. A Japanese name (703) and an attribute (704) are acquired and outputted from the software components of a dictionary based on a data item name and it about the data item on the input medium which becomes the information origin at the time of editing an output item about the input subject name (1812) and attribute (1813) of the input item column.
[0108] It outputs to the keyword column (1809), the edit approach column (1810), and the edit partition column (1811) based on the content defined as the input edit definition part (707) and output edit definition part (708) of a dictionary by the column of edit. Since the edit approach of the software components of a dictionary is described by the keyword about the edit approach column (1810), it outputs by performing keyword Japanese conversion. When the edit approach is developed from input edit of the software components of a dictionary by the edit partition column (1811) and it is developed from "input edit" and output edit by it, it outputs to it with "output edit."
[0109] The criteria at the time of developing the edit approach defined as the software components of the dictionary to a data item here to the keyword column (1809) and the edit approach column (1810) are explained.
[0110] First, the expansion criteria in input edit are shown. Although it is not included in the input medium of this program that this data item is newly generated based on the data in an input medium

during that the input media to this program are media into which it is inputted by the user, such as an input screen or an input document, or this program execution, i.e., this data item, it is two points of being contained in an output media, and input edit will be developed if the method of either 1 is filled.
[0111] The expansion criteria in output edit are shown. The output medias of this program are media, such as an output screen or a document, and output edit is developed when this data item is included in both an input medium and an output media.
[0112] Hereafter, the example of generation of the edit condition table in this example is shown in a detail. Since the groups of the output media defined as the program definition-part article and an input medium are an output document "KYLST" (603), an input file "KYFIL" (601) and an output document "KYERR" (604), and an input file "KYFIL" (601), its attention is first paid to the 1st set of "KYLST(s)" and "KYFIL(s)." A document definition-part article (drawing 13 (a)) is inputted based on "KYLST", and its attention is paid to the data item defined there.
[0113] Since the data item defined as the 1st of a data item name (911) is "SIMEI-NO", from the software components (drawing 7) of a dictionary, a document definition-part article to an attribute "X (4)" (912) is acquired, and it outputs the Japanese name "a name number" (703) of a data item. Since "SIMEI-NO" is contained in an input medium, input edit is not developed. Next, nothing is defined although the output edit definition part of a dictionary is referred to. This shows that it outputs at the time of output edit, without also performing processing of what about this this data item. Then, it outputs, saying a keyword is made into a null and "is transmitted as it is" to the edit approach column (1810).
[0114] Since the data item defined as the 2nd of a data item name (911) is "SIMEI", similarly, from the software components (drawing 8) of a dictionary, a document definition-part name of article to an attribute "N (7)" (912) is acquired, and it outputs the Japanese name "a name" of a data item. Next, with reference to the input edit definition part (707) of a dictionary, the keyword "SUB (SUB02, SIMEI-NO)" of the edit approach is acquired. This is the keyword which shows the software components of a subroutine call, and it is shown that "SUB02" is an input parameter to which "SIMEI-NO" hands over a subroutine name to the subroutine. Therefore, it turns out that the input subject name (1812) to "SIMEI" is "SIMEI-NO."
[0115] Since "SIMEI" is not contained in "KYFIL" which is the input medium of this program, the expansion criteria in input edit are fulfilled. Therefore, an input item "SIMEI-NO" and its attribute are outputted to the column (1809 1810) of edit of what performed a keyword "SUB" and keyword Japanese transform processing at the input item column (1812 1813), respectively.
[0116] In addition, since nothing is defined as the output edit definition part of a dictionary in this example, only input edit is developed in this way, but when further defined also as the output edit definition part, both input edit and output edit are developed and outputted to this sequence.
[0117] Since the data item defined as the 3rd of a data item name (911) is "NYUSYA-YMD", it acquires the software components (drawing 9) of a dictionary to an "entrance date", and outputs a document definition-part name of article to an attribute "X (8)" (912) for it to the output item column (1807 1808), respectively. Next, a keyword "DATE" is acquired with reference to the output edit definition part (708) of a dictionary. Since this output media is a document, the expansion criteria in output edit are fulfilled. Then, it outputs "it changing into the format of YY.MM.DD" to the keyword column (1809) and the edit approach column (1810), and "NYUSYA-YMD" and the attribute of self and a same name are acquired in the input item column (1812 1813) from the software components of a dictionary, and are outputted to it, respectively. [which performed keyword Japanese conversion with the keyword]
[0118] About the data item "SYOKUI-CD" defined as the 4th of a data item name (911), it develops like the case of "SIMEI-NO."
[0119] About the data item "SIKYU-GK" defined as the 5th of a data item name (911) Although there is no definition information in the input edit definition part of a dictionary, and an output edit definition part like "SIMEI-NO" The attributes acquired from the document definition-part article are "Z, ZZZ, ZZ9" (912), and since this performs zero suppress and comma edit, it outputs to the edit approach column (1810) in this case, saying "It edits into the format of Z, ZZZ, and ZZ9."

[0120] Hereafter, the procedure of program generation is explained.
[0121] In advance of explanation of the procedure of program generation, the storing format of the program specification on a program specification library (210) is explained using drawing 23 - drawing 26 .
[0122] Drawing 23 is the storing format of the program manipulation schematic diagram on a program specification library (210). This format is constituted by the column of the program name (1901) of a user program, a symbolic name (1902), the standard-pattern name (1903) used by the user program, the Japanese name (1904) of input/output media, a definition-part name of article (1905), an I/O partition (1906), an activity partition (1907), a logical unit name (1908), a prefix (1909), and the processing outline (1910) of a program. This is constituted by the completely same data as the example of an output of the program manipulation schematic diagram in drawing 18 .
[0123] Drawing 24 is the storing format of the check condition table on a program specification library (210). This format is constituted by the column of the definition-part name of article (2001) of an input medium, the definition-part name of article (2002) of an output media, the Japanese name (2003) of an input item, a symbolic name (2004), an attribute (2005), the keyword (2006) of check processing, check conditions (2007), an error code (2008), and an error message (2009). This is constituted by the completely same data as the example of an output of the check condition table in drawing 20 .
[0124] Drawing 25 is the storing format of the related check condition table on a program specification library (210). This format is constituted by the column of the keyword (2104) of the definition-part name of article (2101) of input/output media, the Japanese name (2102) of an input item, a symbolic name (2103), and check conditions, an error code (2105), an error message (2106), and a matrix (2107). This is constituted by the completely same data as the example of an output of the related check condition table in drawing 21 .
[0125] Drawing 26 is the storing format of the edit condition table on a program specification library (210). This format is constituted by the column of the keyword (2206) of the definition-part name of article (2201) of an input medium, the definition-part name of article (2202) of an output media, the Japanese name (2203) of an output item, a symbolic name (2204), an attribute (2205), and the edit approach, the edit approach (2207), an edit partition (2208), the Japanese name (2209) of an input item, a symbolic name (2210), and an attribute (2211). This is constituted by the completely same data as the example of an output of the edit condition table in drawing 22 .
[0126] Next, the procedure of program generation is explained based on a flow chart.
[0127] Drawing 4 is a flow chart which shows the procedure of program generation, and drawing 27 - drawing 29 are the examples of expansion of the source program by the program generation function based on the definition-part article (drawing 13 - drawing 14) in program specification (drawing 23 - drawing 26) and a data definition document. Hereafter, the concrete procedure of a program generation function is explained using this example.
[0128] A generation function acquires the standard-pattern name "CHK01" (1903) first used from the program manipulation schematic diagram (1900) of the user program for generation, and a corresponding standard pattern (1202) is inputted from a standard-pattern library (205) (step 401).
[0129] The definition-part article of input/output media is inputted from a data definition document (203) based on the input/output-media name which the I/O bill of materials (402) defined with the program manipulation schematic diagram of this program, the source code of the I/O declaratives of this program is generated, and it compounds to the standard pattern which inputted this at step 401.
[0130] The expansion approach of an I/O declaratives is concretely explained using drawing 27 . A program generation function from the input/output-media definition-part name of article (1905) indicated by the program manipulation schematic diagram (1900) "KYFIL", Acquire definition-part names of articles, such as "KYLST", and define the file components (1001) and a document definition-part article (901 907) are read. [/ based on this] An I/O environmental definition (2302 2303), the file description (2304 2306), and the data description (2305 2307) of this program are generated from such information, and this is compounded to a standard pattern (1202).
[0131] Document printing processing expansion (404) is compounded to the standard pattern which

carried out when this program was using the standard pattern which outputs a document, inputted the document definition-part article which corresponds from a data definition document (203) based on the output-media definition-part name of article indicated by the program manipulation schematic diagram of this program, from now on, generated the working area for document edit, and the document output procedure, and was inputted at step 401.

[0132] Document printing processing expansion processing (404) is concretely explained using drawing 27 . A program generation function reads a document definition-part article (901 907) based on the document definition-part name of article (1905) indicated by the program manipulation schematic diagram (1900), generates working areas (2308), such as a counter used from these contents of a definition at the time of the working area (2309) for document edit about a document, and program execution, and compounds this to a standard pattern (1202).

[0133] In a check, a related check, and edit processing expansion (405), based on the keyword indicated by the check condition table of the program specification, the related check condition table, and the edit condition table, processing based on data items is generated and this is developed, respectively to the check processing section (1203) of a standard pattern, the related check processing section (1204), and the edit processing section (1205).

[0134] The source program expansion approach is concretely explained taking the case of the expansion from an edit condition table using drawing 29 . The edit condition table (2200) of this program is read first, the processing to this data item is generated based on the edit conditions defined as this edit condition table by the keyword (2206), and this is developed in the edit processing section (1205) of a standard pattern, respectively. It generates simultaneously also about a working area required for a data item editing in that case. For example, in the case of the keyword "DATE" (2206) of the 3rd line of an edit condition table (2200), as shown in a field (2310), the working area for dividing the date into a year, the moon, and a day and the working area for edit are generated.

[0135] In case the processing to a data item is furthermore developed, the prefix (1909) which the program manipulation schematic diagram (1900) defined as this example showed is added to the subject name on each file. For example, as shown in a statement (2314), to the data item "SIKYU-GK", "IN -" was added to the input item, "OU -" was added to the output item, respectively, and the notation subject name on an input file has determined the notation subject name on "IN-SIKYU-GK" (2315) and an output file as "OU-SIKYU-GK" (2316) and a program generate time. The software components based on data items are made to become independent of the definition-part article of input/output media, such as a file treating that data item, by this.

[0136] In addition, drawing 28 replaces the file symbolic name of the undefined in a standard pattern (1202) by the file symbolic name (1004) of define the file components.

[0137] With a source program output (406), the user program of the object currently generated by the above processing is outputted to a source program library (211).

[0138] In addition, in this example, although the example of the program of batch processing was given and explained, it is generable similarly about the program which performs access to the program of interactive processing accompanied by access to a database, the resource on other equipments, or a process.

[0139] Moreover, in this example, although the source program is generated from program specification, it is also possible to input a dictionary (201), a related check definition document (202), a data definition document (203), a program definition document (204), and a standard function explanation library (206), without minding program specification, and to generate a direct source program, and also when using this approach, it is contained in the range of this invention. In this case, it is the same point as a program specification generate time, and the software components based on data items are chosen based on the data item which exists in the input/output media of the target user program, and it compounds to a standard pattern.

[0140] It doubles and the program specification edit function (212) and software components reverse generation function (208) in this example are explained to the last, while the example of an activity of these functions is shown. The example of an activity to be introduced from now on is an example which

makes program specification correction using a program specification edit function (212), and corrects software components from the program specification after correction to the example program shown in drawing 6 using a software components reverse generation function (208).
[0141] It introduces about the example of correction of various program specification first. Drawing 24 (a) is the example of the check condition table immediately after generation, and drawing 24 (b) is the example of the check condition table after correction. Drawing 25 (a) is the example of the related check condition table immediately after generation, and drawing 25 (b) is the example of the related check condition table after correction. Drawing 26 (a) is the example of the edit condition table immediately after generation, and drawing 26 (b) is the example of the edit condition table after correction.
[0142] Here, an edit condition table (drawing 26) is taken up and the content of correction is introduced to a detail. In the 3rd line (2222) of drawing 26 (b), the edit approach was changed [at it] into the null with the keyword "it transmitting as it is" from "format of YY.MM.DD --" from "DATE" (this actuation is considered as correction 1). This means not performing processing of what to data, either to a data item "NYUSYA-YMD" at the time of output edit. Moreover, in drawing 26 (b), the 4th line (2212) in drawing 26 (a) was deleted (it considers as this correction 2). This means deleting a data item "SYOKUI-CD" from an output media. In the 4th line (2223) of drawing 26 (b), the attribute of an output item "SIKYU-GK" was changed into "Z, ZZ9" from "Z, ZZZ, ZZ9." Furthermore, it was presupposed from the null the edit approach "is edited into the format of Z and ZZ9 per 1000 yen" with a keyword at it at "COMP (SIKYU-GK/1000)" (it considers as this correction 3). In case this outputs "SIKYU-GK", it newly defines output edit processing in which the value divided by 1000 is outputted.
[0143] In addition, about correction of the program specification in a program specification library (210), it can also carry out using the general-purpose software for spreadsheets, or the software for document preparation, without using an exclusive editor, since it stores in the image of a simple table as program specification is shown in drawing 23 - drawing 26 , This is also contained in the range of this invention. Furthermore, it transmits on the equipment group B (118) which is equipment different from the equipment group A (108) which generated program specification, and it is also possible on the equipment group B to perform edit using the general-purpose software for spreadsheets or the software for document preparation, and this is also contained in the range of this invention.
[0144] Next, the procedure of software components reverse generation is explained using drawing 5 . The reverse generation said by this example means rewriting the software components which are the information origin on this based on the program specification with which correction was added, and making the content of correction of program specification reflect in it.
[0145] In the software components reverse generation function (208) of this example, a program manipulation processing schematic diagram is inputted (501), and reverse generation is carried out for a program definition document (204) (502). Next, a check condition table is inputted (503) and reverse generation is performed for a dictionary (201) and a data definition document (203) (504). Next, a related check condition table is inputted (505) and reverse generation is performed for a dictionary (201), a related check definition document (202), and a data definition document (203) (506). Next, an edit condition table is inputted (507) and reverse generation is performed for a dictionary (201) and a data definition document (203) (508).
[0146] Next, a reverse generation result is explained. When reverse generation is performed based on the program specification after the above-mentioned correction, only what had modification in the content of a definition a reverse generation front and after reverse generation is shown in drawing 30 - drawing 36 . Drawing 30 - drawing 34 are the examples of the software components of the dictionary after reverse generation, and the examples of a definition before reverse generation are drawing 7 - drawing 11 . Drawing 35 is the example of the related check definition-part article after reverse generation, and the example of a definition before reverse generation is drawing 12 . Drawing 36 is the example of the document definition-part article to "KYLST" among the data definition documents after reverse generation, and the example of a definition before reverse generation of this is drawing 13 (a).
[0147] The concrete procedure of reverse generation is explained based on the example of an edit condition table [finishing / the above-mentioned correction] (drawing 26 (b)).

[0148] The keyword (2214) is changed into the null in correction 1. In this case, first, with reference to an edit partition (2215), since this is "output edit", a reverse generation function acquires an input item "NYUSYA-YMD" (2216) with reference to a symbolic name (2210). The software components of the dictionary (201) corresponding to it are set as the object of reverse generation. The keyword is changed into the null as shown in the input edit definition part (2401) of the software components (drawing 32) of the dictionary after reverse generation.
[0149] In drawing 26 (b), the data item "SYOKUI-CD" (2212) was deleted from the output item by correction 2. In this case, a reverse generation function deletes "SYOKUI-CD" from the document definition-part article "KYLST" (drawing 13 (a)) of the document which is the output media of this program. In the document definition-part article after reverse generation (drawing 36), the data item "SYOKUI-CD" is deleted as shown in the document layout definition part (2601).
[0150] In correction 3, the keyword is changed like correction 1. Also in this case, since an edit partition is "output edit" (2220), it acquires a symbolic name "SIKYU-GK" (2221), and the output edit definition part of the software components (drawing 34) of the dictionary corresponding to it is set as the object of reverse generation. Since the keyword (2206) after correction is "COMP (SIKYU-GK/1000)" (2219), it updates the keyword (2402) of output edit of the software components of the dictionary to "SIKYU-GK."
[0151] Moreover, in correction 3, the attribute (2218) of a symbolic name "SIKYU-GK" (2217) is changed further. In this case, this data item attribute in the document definition-part article "KYLST" (drawing 13 (a)) of the document which is an output media is changed. The content of modification is reflected as shown in (2602) of the document layout definition part of the document definition-part article after reverse generation (drawing 36).
[0152] Thus, with the procedure of a program specification generation function (209), a software components reverse generation function (208) specifies software components with correction from a program specification library (210) in the procedure of reverse, and updates software components based on the content of the program specification library (210). Reverse generation of software components can be performed like an edit condition table also about the case where it carries out based on a program manipulation schematic diagram, a check condition table, and a related check condition table.
[0153]
[Effect of the Invention] Since this invention has the procedure explained above, it has effectiveness which is indicated below.
[0154] We defined related check processing in which divided independent processing of as opposed to each data item for software into some classes, and defined it as a data item unit as software components, and the validity of data was judged based on the correlation between two or more data items, as software components, and decided to manage software ****** based on these data based on a data item name. On the other hand, about the software components which are not contained in the software components based on these data The software components which defined the file, an attribute, a layout of a document, etc., The software components, screen which defined the input/output-media information on a program unit, Pay one's attention to the special feature of a control structure, and processing programs, such as a document and a file, are divided into the class of shoes. We decided to divide and define operation individual processing as the software components called the program skeleton standard pattern which it does not have in itself, and the software components called the standard function recital article which met program skeleton standard-pattern components.
[0155] By the above, about the division approach of the software for components-ization in components-izing of software The clear fixed criteria for dividing software focusing on a data item can be established. Since the approach of managing in the format which agreed on these criteria by furthermore storing these software components in the file which had specific structure, respectively is establishable, The components of software are standardized, and it becomes possible to make reuse of the existing software easy, as a result the productivity drive of software can be planned.
[0156] Furthermore, the above-mentioned dictionary, a related check definition document, a data definition document, Software components are chosen from a program definition document and a

standard function explanation library based on the data item in the input/output media of the user program made into an object. The program specification which consists of some kinds of specification is generated automatically. Based on program specification Program skeleton standard-pattern components, It considered as the method which generates a source program automatically by choosing and compounding the software components in a dictionary, and software components related check definition in the letter.

[0157] Therefore, it becomes possible to be able to generate a source program automatically, without an architect or a programmer performing matching with a user program and software components, to automate a program specification creation activity, and to realize automatic succession of the design information from the design process of software components to a program-design process. The productivity and dependability of software can be improved by this.

[0158] In addition, in order to attain components-ization of the above software under still stricter standardization criteria, in this invention, it decided to divide the independent processing to the data item in a dictionary into three kinds, a check, input edit, and output edit. It becomes possible to limit further the structure of the software components stored in a dictionary by this, and the productivity and dependability of software components improve, therefore reuse also becomes easy.

[0159] Moreover, in this invention, we enabled correction of modification and the addition to the generated program specification, deletion, etc., and decided to generate a source program automatically based on the program specification modified here.

[0160] By this, it becomes possible, also when a different specification from the software components beforehand prepared for program specification for some reasons of the program for generation including exception handling is described to make this reflect in source program generation, and the productivity drive of software and improvement in dependability can be aimed at.

[0161] Moreover, in this invention, it made it possible to correct the content of a definition of the software components automatically stored in the dictionary, the related check definition document, the data definition document, and the program definition document from the corrected program specification. For this reason, correction of the software components accompanying specification modification etc. can be automated, and the productivity of software can also aim at improvement with the improvement in dependability of software components.

[0162] Furthermore by this invention, it made it possible to correct program specification on the equipment different from the equipment which generated program specification using general-purpose software to the generated program specification. even if it comes out without being restricted to the employment conditions of specific equipment to a software-development person, while aiming at reduction of the plant-and-equipment investment for using the software development tool in this invention by this, and it does not learn the operating instructions of a tool proper, it can become possible to offer the work environment which can perform a software development using the general-purpose software which already got used and was familiar, as a result the productivity of software can be raised.

[Translation done.]

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

TECHNICAL FIELD

[Industrial Application] This invention relates to the suitable software-development exchange approach to increase the efficiency of a new software development by starting the productivity drive of software, especially components-izing the existing software, and generating program specification automatically from these components, and generating a program automatically from program specification.

[Translation done.]

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

PRIOR ART

[Description of the Prior Art] There is a method of reusing the software of existing [one] of the leading approach for improving software productivity, and developing new software. When reusing this existing software and developing new software, it is common to reuse in the phase of operation system creation or the phase of a programming, and to correct by analyzing the document and source program with which the programmer described the specification of the existing software.
[0003] However, also in the software of operation of the same kind, in order to add the correction accompanying a difference of two or more users' requirement specification to one software with it difficult [for the requirement specification to be various and to pinpoint a correction part only from a document or a source program by the approach of the above reuse] by the user, there was a fault of software maintainability deteriorating.
[0004] For such a fault, reuse of the existing software was given up, software was completely newly developed even in the software development of operation of the same kind, in many cases, and the productivity drive was barred.
[0005] By subdividing in that function as one approach of solving this problem paying attention to software, it components-izes and there is a method of generating a new program automatically by defining matching with these components and programs, respectively. When dividing software into a small-scale components group by this approach and developing new software, it becomes possible to reuse the existing software on components level, and could respond by choice of components, modification, and addition to the difference in the application of software, and the difference in requirement specification.
[0006] In addition, the pages 19-24 "system development support-software "EAGLE"" of Hitachi criticism VOL.66 (March, 1984 issuance) etc. are one of those which are related as this kind of a technique.

[Translation done.]

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

EFFECT OF THE INVENTION

[Effect of the Invention] Since this invention has the procedure explained above, it has effectiveness which is indicated below.
[0154] We defined related check processing in which divided independent processing of as opposed to each data item for software into some classes, and defined it as a data item unit as software components, and the validity of data was judged based on the correlation between two or more data items, as software components, and decided to manage software ****** based on these data based on a data item name. On the other hand, they are a file, an attribute, a layout of a document, etc. about the software components which are not contained in the software components based on these data. We decided to pay one's attention to the special feature of a control structure, to divide into the class of shoes processing programs, such as defined software components, software components which defined the input/output-media information on a program unit, a screen, a document, and a file, to divide into the software components called the program skeleton standard pattern which does not have operation individual processing in itself, and the software components called the standard function recital article which met program skeleton standard-pattern components, and to give a definition.
[0155] Center on a data item about the division approach of the software for components-ization in components-izing of software by the above. The clear fixed criteria for dividing software can be established, since the approach of managing in the format which agreed on these criteria by storing these software components in the file which had specific structure, respectively further is establishable, the components of software are standardized and it becomes possible to make reuse of the existing software easy, as a result the productivity drive of software can be planned.
[0156] Furthermore, the thing for which software components choose from the above-mentioned dictionary, a related check definition document, a data definition document, a program definition document, and a standard function explanation library based on the data item in the input/output media of the user program made into an object, the program specification which consists of some kinds of specification generates automatically, and program skeleton standard-pattern components, and the software components in a dictionary and software components related check definition in the letter choose and compound based on program specification It considered as the method which generates a source program automatically.
[0157] Therefore, it becomes possible to be able to generate a source program automatically, without an architect or a programmer performing matching with a user program and software components, to automate a program specification creation activity, and to realize automatic succession of the design information from the design process of software components to a program-design process. The productivity and dependability of software can be improved by this.
[0158] In addition, in order to attain components-ization of the above software under still stricter standardization criteria, in this invention, it decided to divide the independent processing to the data item in a dictionary into three kinds, a check, input edit, and output edit. It becomes possible to limit further the structure of the software components stored in a dictionary by this, and the productivity and dependability of software components improve, therefore reuse also becomes easy.

[0159] Moreover, in this invention, we enabled correction of modification and the addition to the generated program specification, deletion, etc., and decided to generate a source program automatically based on the program specification modified here.
[0160] By this, it becomes possible, also when a different specification from the software components beforehand prepared for program specification for some reasons of the program for generation including exception handling is described to make this reflect in source program generation, and the productivity drive of software and improvement in dependability can be aimed at.
[0161] Moreover, in this invention, it made it possible to correct the content of a definition of the software components automatically stored in the dictionary, the related check definition document, the data definition document, and the program definition document from the corrected program specification. For this reason, correction of the software components accompanying specification modification etc. can be automated, and the productivity of software can also aim at improvement with the improvement in dependability of software components.
[0162] Furthermore by this invention, it made it possible to correct program specification on the equipment different from the equipment which generated program specification using general-purpose software to the generated program specification. even if it comes out without being restricted to the employment conditions of specific equipment to a software-development person, while aiming at reduction of the plant-and-equipment investment for using the software development tool in this invention by this, and it does not learn the operating instructions of a tool proper, it can become possible to offer the work environment which can perform a software development using the general-purpose software which already got used and was familiar, as a result the productivity of software can be raised.

[Translation done.]

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

TECHNICAL PROBLEM

[Problem(s) to be Solved by the Invention] The following technical problems occur in the conventional approach.
[0008] The 1st technical problem in the conventional technique is in the succession approach to the means of components-izing of software itself, and the program specification document of software components design information.
[0009] Since fragmentation of a function was performing components-ization of software, in case components-ization was performed with the conventional technique of generating a program automatically by components-izing the above-mentioned software and compounding it, it was difficult in what kind of viewpoint to subdivide a function or which level to use as one component, and to establish criteria clear about ********, and the division approach of software was various by the architect of components. For this reason, it was difficult to put a standardization of software components into practice.
[0010] Furthermore, since an architect or a programmer had to program 11 define matching with the target user program and components in order to generate a source program, a productivity drive and improvement in dependability were not fully able to be aimed at.
[0011] It was that by which most creation of a program specification document, on the other hand, depends creation of these software components on human being's independent handicraft although some means to generate them to the means for creating software components and a source program were developed. Although there is the approach of managing as an electronic filing document which created the program specification document using computers only for document preparation, such as the so-called word processor, and fair copy and storage of a document can be supported as for this, the program-design activity itself is still based on a program-design person's handicraft including the electronic-filing-document creation activity. It is impossible to realize automatic succession of the design information from the design process of software components to a program-design process by such creation approach of a program specification document. For this reason, since check that it is difficult and the source program and program specification document which are further generated from software components and it are in agreement is also left to the handicraft of a programmer, the productivity drive of a program specification document creation activity also serves as hindrance of the improvement in software reliability.
[0012] The 2nd technical problem in the conventional technique is not to have a means to generate the source program as a program specification document, when a different specification from the software components beforehand prepared for the program specification document is described.
[0013] Even if it described a different specification from the software components beforehand prepared for the program specification document for some reasons of doing most program specification document creation activities by handicraft in the conventional technique as the 1st technical problem described, and the program for generation including exception handling for this reason, it was impossible to have made this reflect in source program generation. Therefore, since it was necessary to correct the source program with which the programmer was generated according to a program specification document in

such a case, a productivity drive and improvement in dependability were not fully able to be aimed at.
[0014] When modification arises in the specification of an object user program, the 3rd technical problem in the conventional technique is that software components are automatically uncorrectable according to the specification described by the program specification document, even if it corrects a program specification document. Moreover, it is also the same as when the software components itself are the right specifications of original [specification / which the specification has mistaken by the poor definition and was described by the program specification document] as another case. Therefore, since it was necessary to correct the source program with which the programmer was generated according to a program specification document in such a case, and to also make correction of software components further, it became the hindrance of the improvement in dependability of software components, as a result had also become the hindrance of software components reuse.
[0015] The 4th technical problem in the conventional technique is in the employment approach and operability of the software which supports development by components-izing of such software.
[0016] if those who can perform it only by the actuation for using the software (such software -- a following software development tool -- or it only being called a tool) which supports a software development which was described until now using the equipment with which the tool concerned was incorporated, and use the tool did not follow the operating instructions which the tool surely defined, when there were, they did not become. [no] For this reason, it is necessary to learn the operating instructions of the proper which a software-development person's activity was restricted to an installation, an employment time zone, etc. of equipment when the tool was incorporated on the other hand by a manager having to perform plant-and-equipment investment corresponding to a software-development person's manpower, and the tool defined, as a result has become the hindrance of a software productivity drive.
[0017] By establishing the fixed criteria for which it does not depend on an architect's individuality or the special feature of application in components-izing of software, and establishing the approach of managing components in the format corresponding to these criteria further, the 1st object of this invention standardizes the components of software, and is to make reuse of the existing software easy.
[0018] Furthermore, it is in inheriting the design information of software components automatically to program specification and a source program, and improving the productivity and dependability of software by making it possible to choose software components required for the program concerned based on the name of a data item, to generate program specification automatically, and to generate a source program automatically from this program specification, without an architect or a programmer defining matching with a user program and software components.
[0019] The 2nd object of this invention is by enabling correction of modification, an addition, deletion, etc. to the generated program specification, and making it possible to generate the source program as program specification automatically, also when the specification indicated by the program specification after correction differs from the specification of the software components prepared beforehand to improve the productivity and dependability of software.
[0020] The 3rd object of this invention is by keeping the consistency of program specification and software components automatic, and improving the dependability of software components, as a result making reuse easy by enabling correction of program specification as mentioned above, and making it possible to correct software components automatically according to the specification indicated by the program specification after correction if needed to also raise the productivity of software.
[0021] Modification to the program specification which stated the 4th object of this invention for the 2nd object, Transfer methods, such as an addition and deletion, by so-called personal computers and workstations other than the specific equipment with which the software development tool concerned was incorporated By making it possible to carry out using the software for the so-called spreadsheets of a general purpose, or the software for document preparation Without being restricted to the employment conditions of the equipment of specification [a software-development person], while aiming at reduction of the plant-and-equipment investment for using the software development tool concerned And it is in making it possible to offer the work environment which can perform a software development

using the general-purpose software which already got used and was familiar, even if it does not learn the operating instructions of a tool proper, as a result raising the productivity of software.

[Translation done.]

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

MEANS

[Means for Solving the Problem] In order to attain the 1st object, by this invention, divide independent processing of as opposed to each data item for software into some classes, and it is defined as a data item unit as software components. Related check processing in which the validity of data is judged based on the correlation between two or more data items to be a means to store in the file called a dictionary is defined as software components. It is characterized by having a means to store in the file called a related check definition document, and a means to manage the software components stored in these dictionaries and a related check definition document based on a data item name.
[0023] The processing hereafter defined in the software components based on data items, and calls and these components in accordance with the software components stored in the above-mentioned dictionary and a related check definition document is called processing based on data items.
[0024] Furthermore, a means to store in the file which defines information other than a means to define the software components based on [above-mentioned] data items, such as a file, and the attribute of a document, a layout, and is called a data definition document, A means to store in the file which defines the information on input/output media as a program unit, and is called a program definition document, Pay one's attention to the special feature of a control structure, and processing programs, such as a screen, a document, and a file, are divided into the class of shoes. A means to store in the file which components-izes as a program skeleton standard pattern which does not have operation individual processing in itself, and is called a standard-pattern library, It is characterized by having a means to store the standard function recital article which met program skeleton standard-pattern components in the file called a standard function explanation library.
[0025] Furthermore, the above-mentioned dictionary, a related check definition document, a data definition document, Software components are chosen from a program definition document and a standard function explanation library based on the data item in the input/output media of the user program made into an object. A means to generate automatically the program specification which consists of some kinds of specification, It is characterized by having a means to generate a source program automatically by choosing and compounding program skeleton standard-pattern components, and the software components in a dictionary and software components related check definition in the letter based on program specification.
[0026] As a means for attaining the 1st object under still stricter standardization criteria, it is characterized by having the means which divides the independent processing to the data item in a dictionary into three kinds, a check, input edit, and output edit, by this invention.
[0027] In order to attain the 2nd object, in this invention, it is characterized by having the means which enables correction of modification and the addition to the generated program specification, deletion, etc., and a means to generate a source program automatically based on the program specification modified here.
[0028] In order to attain the 3rd object, in this invention, it is characterized by having a means to correct the content of a definition of the software components automatically stored in the dictionary, the related check definition document, the data definition document, and the program definition document from the

corrected program specification.
[0029] In order to attain the 4th object, in this invention, it is characterized by having ****** which makes it possible to correct program specification to the generated program specification on the equipment different from the equipment which generated program specification using general-purpose software.

[Translation done.]

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

OPERATION

[Function] In order to attain the 1st object, we defined related check processing in which divided independent processing of as opposed to each data item for software into some classes, and defined it as a data item unit as software components by this invention, and the validity of data was judged based on the correlation between two or more data items, as software components, and decided to manage software ****** based on these data based on a data item name. On the other hand, they are a file, an attribute, a layout of a document, etc. about the software components which are not contained in the software components based on these data. We decided to pay one's attention to the special feature of a control structure, to divide into the class of shoes processing programs, such as defined software components, software components which defined the input/output-media information on a program unit, a screen, a document, and a file, to divide into the software components called the program skeleton standard pattern which does not have operation individual processing in itself, and the software components called the standard function recital article which met program skeleton standard-pattern components, and to give a definition.
[0031] The clear fixed criteria of dividing software focusing on a data item in components-izing of software by the above about the division approach of the software for [the conventional technique / with an architect's individuality or the special feature of application] various components-ization can be established. Since the approach of managing in the format which agreed on these criteria by furthermore storing these software components in the file which had specific structure, respectively is establishable, the components of software can be standardized, and it can become possible to make reuse of the existing software easy, as a result the productivity drive of software can be planned, and the technical problem of the conventional technique can be solved.
[0032] Furthermore, the thing for which software components choose from the above-mentioned dictionary, a related check definition document, a data definition document, a program definition document, and a standard function explanation library based on the data item in the input/output media of the user program made into an object, the program specification which consists of some kinds of specification generates automatically, and program skeleton standard-pattern components, and the software components in a dictionary and software components related check definition in the letter choose and compound based on program specification It considered as the method which generates a source program automatically.
[0033] Therefore, in a Prior art, it becomes possible to be able to generate a source program automatically, to automate the program specification creation activity for which most depended on an architect's handicraft in the Prior art, and to realize automatic succession of the design information from the design process of software components to a program-design process, without an architect or a programmer performing matching with the program and software components which had to define one 1 user program. By this, the productivity and dependability of software can be improved and the technical problem of the conventional technique can be solved.
[0034] As a means for attaining the 1st object under still stricter standardization criteria, it decided to divide the independent processing to the data item in a dictionary into three kinds, a check, input edit,

and output edit, by this invention. It becomes possible to limit further the structure of the software components stored in a dictionary by this, and the productivity and dependability of software components improve, therefore reuse also becomes easy.
[0035] Since the 2nd object is attained, in this invention, correction of modification and the addition to the generated program specification, deletion, etc. is enabled, and a source program can be generated automatically based on the program specification modified here.
[0036] Some reasons of the program for generation including exception handling by this, Since it becomes possible to make this reflect in source program generation also when a different specification from the software components beforehand prepared for program specification is described, it becomes unnecessary to correct the source program generated like the conventional technique as software components according to a program specification document, the productivity drive of software and improvement in dependability can be aimed at, and the technical problem of the conventional technique can be solved.
[0037] In order to attain the 3rd object, in this invention, it made it possible to correct the content of a definition of the software components automatically stored in the dictionary, the related check definition document, the data definition document, and the program definition document from the corrected program specification. Like the case stated with the 3rd technical problem in the conventional technique, for the reason of modification occurring in the specification of an object user program, also when program specification is corrected, in accordance with the specification described with program specification, it can correct to software components automatically. For this reason, in such a case, with the conventional technique, the correction of software components which was being done manually can be automated, the productivity of software also improves with the improvement in dependability of software components, and the technical problem of the conventional technique can be solved.
[0038] In order to attain the 4th object, in this invention, it made it possible to correct program specification to the generated program specification on the equipment different from the equipment which generated program specification using general-purpose software. It becomes possible to perform transfer methods, such as modification and the addition to program specification, and deletion, using the software for the so-called spreadsheets of a general purpose, or the software for document preparation by so-called personal computers and workstations other than the specific equipment with which the software development tool concerned was incorporated by this. Therefore, while aiming at reduction of the plant-and-equipment investment for using the software development tool concerned, without being restricted to the employment conditions of specific equipment to a software-development person, and even if it do not learn the operating instructions of a tool proper, it can become possible to offer the work environment which can perform a software development using the general-purpose software which already got used and be familiar, as a result the productivity of software can be raised, and the technical problem of the conventional technique can be solved.

[Translation done.]

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

EXAMPLE

[Example] Hereafter, the example of this invention is explained to a detail using drawing.
[0040] First, the means for realizing the hardware configuration for realizing this example using drawing 1 and drawing 2 and some functions which this example has is explained.
[0041] Drawing 1 shows the hardware configuration of the system for realizing this invention. The equipment group A (108) is an equipment group for updating the software components which are generation of program specification or a source program, and the generator information on the program specification from program specification that the generated program specification was corrected and revised (reverse generation being called in this example). Moreover, the equipment group B (118) is used when making correction of program specification on a different equipment group from the equipment group A. The optimal thing is chosen from the environment where the user was placed although it is desirable as an equipment group B to use the so-called personal computer and the so-called workstation. moreover, it is also possible to see two or more sets of equipment groups B, and to use them to the equipment group A, -- carrying out -- reverse -- the equipment group B -- it is nothing and it is also possible to perform function concerning this invention and actuation using the equipment group A altogether.
[0042] First, the equipment group A (108) is explained along with drawing 1 . The equipment group A is equipped with CPU (103) as the central equipment. The input unit for performing the data input at the time of editing the program specification which performed the input for ordering it activation of generation, or was generated (105), The display for displaying the data of the generated program specification etc. (104), and in connecting the printer (102) for outputting the generated program specification to CPU as the input/output equipment and also using the equipment group B The communication device (107) for transmitting the generated program specification to the equipment group B, or receiving the program specification edited by the equipment group B is connected to CPU.
[0043] While performing to CPU external storage (106) for storing the software components which serve as the generator again at a program specification generate time, and the program specification and the source program after generation, and generation and edit of program specification, the memory (101) for storing a program and data is connected. The file of the dictionary (201) shown in drawing 2 , a related check definition document (202), a data definition document (203), a program definition document (204), a standard-pattern library (205), a standard function explanation library (206), a program specification library (210), and a source program library (211) is included in the data stored in external storage (106). It is the same as that of the equipment group A also about each equipment of the equipment group B (118). However, the data stored in external storage (116) can be made only into a program specification library (210).
[0044] This example is realized by four functions, the program specification generation function (207) shown in drawing 2 , a program generation function (209), a program specification edit function (212), and a software components reverse generation function (208).
[0045] A program specification generation function (207) is stored in the program specification library (210) which is similarly on external storage (106), or outputs the program specification (213) which

inputted the dictionary (201) on external storage (106), the related check definition document (202), the data definition document (203), the program definition document (204), and the standard function explanation library (206), generated program specification, and was generated to a printer (102).
[0046] A program generation function (209) inputs the program specification library (210), data definition document (203), and standard-pattern library (205) on external storage (106), generates a source program, and stores it in the source program library (211) on external storage (106).
[0047] A program specification edit function (212) is an editor which inputs the program specification library (210) on external storage (106), and corrects addition of program specification, modification, deletion, etc. using an input unit (105) and a display (104).
[0048] A program specification edit function (212) can also be used on the equipment group B (118). In this case, the program specification library (210) of the equipment group A (108) is transmitted to the external storage (116) of the equipment group B from the external storage (106) of the equipment group A, using the display (114) and input unit (115) of the equipment group B, the program specification library (210) on the external storage (116) of the equipment group B is edited, and the corrected program specification is outputted to the printer (112) on the equipment group B.
[0049] The program specification library (210) furthermore edited by the equipment group B can be again transmitted on the external storage (106) of the equipment group A, and the software components reverse generation function (208) stated to a program generation function (209) or a degree by the equipment group A can be used.
[0050] It is also realizable to use the editor of dedication, in order to realize a program specification edit function (212) in more than, and also to carry out using the software for document preparation, the existing general-purpose software, for example, so-called software for spreadsheets, with the conventional technique, and let it be the range of this invention also in this case.
[0051] A software components reverse generation function (208) inputs the program specification library (210) on external storage (106), and updates the software components stored in the dictionary (201) which is similarly on external storage (106), the related check definition document (202), the data definition document (203), and the program definition document (204) if needed.
[0052] In this example, program specification (213) consists of five kinds of specification, a program manipulation schematic diagram (drawing 18), a program function explanatory view (drawing 19), a check condition table (drawing 20), a related check condition table (drawing 21), and an edit condition table (drawing 22).
[0053] In addition, although this example explains five kinds as a case above, besides this, the document relevant to these programs, such as file specification, record specification, document specification, screen specification, database specification, subroutine connection specification, and a resource on other equipments, access specification to a process, can be outputted by the Prior art, and it considers as the range of this invention also in this case.
[0054] In advance of explanation of the concrete procedure of the program specification generation by this invention, the definition-part article stored in a dictionary (201), a related check definition document (202), a data definition document (203), a program definition document (204), a standard-pattern library (205), a standard function explanation library (206), and each library is explained to a detail.
[0055] In addition, drawing 6 is the schematic diagram of the example of a user program made applicable [of this invention] to application. A program 602 reads the input file "KYFIL" (601) which has DS (605), checks validity of data, and normal data are edited into the output document "KYLST" (603) which has a format of a layout 606, respectively, and it outputs them to it at the output document "KYERR" (604) with which unjust data have a format of a layout 607. Henceforth, this example is explained for this example program.
[0056] A dictionary (201) is explained using drawing 7 - drawing 11 . With a dictionary, the software components of the independent processing about each data item are defined and stored in a data item unit. Here, a data item points out the data with professional semantics, such as a name and a date of birth, of a smallest unit. Independent processing is actuation performed only to this data item, and is processing which does not participate in any data items other than itself.

[0057] There are a data item definition part (701), a check processing definition part (706), an input edit definition part (707), and an output edit definition part (708) in the software components of a dictionary. Here, check processing, input edit processing, and output edit processing are defined as follows. That is, the processing which checks the format of data and the validity of data to the input from an input document or an input screen is check processing, the processing changed to the format of storing the inputted data in a database or a file is input edit processing, and the processing change to the format of outputting a database and the data on a file to an output document or an output screen is output edit processing.
[0058] This example shows the example of a definition over the data item of "SIMEI-NO" (drawing 7), "SIMEI" (drawing 8), "NYUSYA-YMD" (drawing 9), "SYOKUI-CD" (drawing 10), and "SIKYU-GK" (drawing 11).
[0059] The attribute which consists of a data item name, a Japanese name, a format, and a digit count, and the comment which specifies the application and semantics of a data item are defined as a data item definition part (701). The data item name was "SIMEI-NO", a Japanese name of (703) is a "name number", (702) defined that the attribute of (704) of data was 9 figures of a figure by the example of drawing 7 , and (705) defines the semantics of a data item by it.
[0060] It is defined as a check processing definition part (706) by the error code and error message which are used in case it tells to a user that unjust things are the conditions for judging that data are unjust. Similarly, it is the keyword which shows that (709) checks by calling a subroutine "SUB01" to the input about this data item, and an error code when (710) is an error is "ER10", and (711) defines that an error message is "not right [a name number]" by the example of drawing 7 .
[0061] An input edit definition part (707) is explained using drawing 8 . This example defines performing input edit about this data item by calling a subroutine "SUB02" by describing it as "SUB (SUB02), SIMEI-NO" to (707). Similarly, about the output edit definition part, by describing it as "DATE" to (708), the example of drawing 9 defines changing and outputting an output form to "YY.MM.DD" (YY: a year, MM:moon, DD:day), in case output edit about this data item is performed. As mentioned above, for the definition of the edit approach of check conditions, and input edit and output edit, it describes using keywords, such as "SUB" and "DATE." In the case of check conditions, in the case of the edit approaches, such as a subroutine call, and a numeric check, an alphabetic-character check, a range check, keywords are a character string showing some common software components, such as a subroutine call, and a formula, character string connection edit, and its syntax rule.
[0062] Next, the related check definition-part article stored in a related check definition document (202) using drawing 12 is explained. A related check definition-part article defines the software components of the related check processing between two or more data items by the format of a decision table.
[0063] This example is an example of the related check definition-part article about a data item "SIKYU-GK" and "SYOKUI-CD." (801) defines that the target data item name is "SIKYU-GK", and (802) defines that a related data item name is "SYOKUI-CD." In addition, of course, there may be two or more related data item names. (803) is the condition definition section of a related check, and (807) is the error-processing definition part of a related check.
[0064] A data item name (804), conditions (805), and a related matrix (806) are defined as the condition definition section (803). An error code (808), an error message (809), and a related matrix (810) are defined as an error-processing definition part (807).
[0065] In addition, conditions (805) are described using a keyword like the software components of a dictionary. this example defines what "SIKYU-GK does not come out size from 80000, it sets an error code to ER50 from 'A03' when SYOKUI-CD is size, and an error message is 'an allowance is below criteria'", and ", as for an error code, it setting to ER60 from 'A03', when SYOKUI-CD is not size, SIKYU-GK being size from 80000 and an error message being 'the allowance is over criteria'."
[0066] Next, a data definition document (203) is explained. A data definition document stores the document definition-part article, define the file components, and record definition components which defined the information on a file or a document. Hereafter, such definition-part articles are explained.
[0067] A document definition-part article is explained using drawing 13 (a). This example defines the

information about the document of the "salary list" (606) in drawing 6 . There are a document attribute definition part (901) and a document layout definition part (907) in a document definition-part article. A document definition-part name of article is "KYLST", (902) defined that a document name of (903) was a "salary list", (905) defines the record length and (906) defines the number of the print lines of the detail business per page. The repeat factor (910) of the initial line (908) when setting forth a data item, an initiation train (909), and a detail line, a data item name (911), and an attribute (912) are defined on a document at a document layout definition part (907).

[0068] Although drawing 13 (b) is similarly a document definition-part article about the "error list" (607) in drawing 6 , since the content is the same as that of drawing 13 (a), explanation is omitted.

[0069] Explanation of define the file components and record definition components is given using drawing 14 .

[0070] This example defines the file of (601) in the example program of drawing 6 , and its record format (605). As for (1002), a define the file components name is "KYFIL", and a file name of (1003) is a "salary file." The file symbolic name which uses (1004) within a program is "KYUYO-FILE." (1005) defined that the record definition components name which defined the record format to this file was "KYREC", and defines the record length (1006), the block length (1007), record format (1008), etc. as an attribute of a file.

[0071] There are a record attribute definition part (1009) and a record layout definition part (1012) in record definition components, and it defines that as for (1010) a record definition components name is "KYREC" and a record name of (1011) is a "salary record" as them. The level number (1013), a data item name (1014), and an attribute (1015) are defined as a record layout definition part.

[0072] In addition, although the program of batch processing is made into the example and the three above-mentioned kinds are used as a data definition document in this example In the system which uses the screen and database other than a file or a document A screen definition-part article, In the case of the system corresponding to a client server model, the resource on other equipments, the access definition-part article to a process, etc. can define a database definition-part article similarly again, and it is contained in this invention also about this.

[0073] A standard-pattern library (205) is explained using drawing 16 .

[0074] Its attention is paid to a control structure, processing programs, such as a screen, a document, and a file, are divided into the class of shoes, and the program skeleton standard-pattern components (it is hereafter called a standard pattern) components-ized as a pattern which does not include operation individual processing in itself are stored. This example is the standard pattern of the type which reads a file, checks and carries out another **** output at normal data and error data. (1201) is a program skeleton part article whose (1202) are a standard-pattern name and is a body of a standard pattern. The check processing section (1203), the related check processing section (1204), and the edit processing section (1205) are processings according to operation individual, and are a part into which these are not included in a standard pattern but a program generation function (209) develops the software components based on data items.

[0075] A standard function explanation library (206) is explained using drawing 17 . With a standard function explanation library, the standard function recital article which met the standard pattern is stored. This example is a standard function recital article corresponding to the standard pattern "CHK01" explained above. (1301) is a standard function recital name of article, and (1302) is description of the processing part of a functional description. In addition, the standard function recital article and the standard pattern have taken the 1 to 1 response, therefore a standard function recital name of article (1301) is a corresponding standard-pattern name and a corresponding homonym.

[0076] The program definition-part article stored in a program definition document (204) is explained using drawing 15 . This example defines the program specification of (602) in the example program of drawing 6 . (1101) is a program attribute definition part and (1106) is an input/output-media definition part. A program symbolic name is "PROG01", a program (1103) Japanese name is "salary list creation", and (1102) defines that the standard-pattern name which uses (1104) by the program is "CHK01." The comment which specifies a program manipulation outline is defined as (1105).

[0077] when using the define the file components name and document definition-part name of article corresponding to input/output media for an input/output-media definition part (1106) (1107) within each I/O partition (it is hereafter called an I/O partition) (1108), an activity partition (1109), a logical unit name (1110), and a program, it adds to a data item -- a prefix (1111) definition is carried out.
[0078] Here, an activity partition (1109) is a partition which shows that it is the file used for a certain special application in each standard pattern. Although the activity partition of input/output media "KYERR" is defined as "ERR" in this example, this shows that it is the output destination change of an error code or an error message edited by check processing in the standard pattern "CHK01."
[0079] Hereafter, the procedure of program specification generation is explained to a detail using the flow chart of drawing 3 . The program definition-part article (shown in drawing 15) of the user program for the introduction generation (it abbreviates to this program below) is inputted (step 301), and the standard-pattern name (1104) currently used by this user program, an input/output-media definition-part name of article (1108), and an activity partition (1109) are acquired.
[0080] Next, a program manipulation schematic diagram is developed and outputted (302).
[0081] The expansion method of a program manipulation schematic diagram is explained using drawing 18 . A program name (1401), a symbolic name (1402), a standard-pattern name (1403), and a processing outline (1412) output what was acquired from the program definition-part article (drawing 15), respectively (1105 (1104 (1102 (1103)))).
[0082] The definition-part name of article (1406) of input/output media (1404), an I/O partition (1407), an activity partition (1408), a logical unit name (1409), and a prefix (1410) output what was acquired from the input/output-media definition part (1111 (1110 (1109 (1108 (1107))))) of a program definition-part article, respectively. The Japanese name (1405) of input/output media (1404) acquires and outputs a file name (1003) and a document name (903) based on the definition-part name of article (1107) of input/output media with reference to define the file components and a document definition-part article.
[0083] in this example, since the definition-part name of article (1107) is defined as "KYFIL" at the program definition-part article, it is alike also in this definition-part name of article, and with reference to define the file components (drawing 14 (a)), a "salary file" is acquired as a file name (1003), and it outputs to a Japanese name (1405). (The processing which acquires the Japanese name of a file or a document with reference to a definition-part article based on a definition-part name of article in this way is henceforth called Japanese Natori profit.) with reference to a dictionary (201), the Japanese name of a data item can be similarly acquired based on a data item name about a data item, and it is called Japanese Natori profit also about this -- based on the further above-mentioned information, it diagrams and the I/O relation of this program is outputted, as shown in a drawing (1411).
[0084] Next, a standard function recital article is inputted based on the standard-pattern name (1104) already acquired from the return program definition-part article to drawing 3 (303), and a program function explanatory view is developed and outputted (304).
[0085] The expansion method of a program function explanatory view is explained using drawing 19 . A program name (1501), a program symbolic name (1502), and a standard-pattern name (1503) output what was acquired from the program definition-part article (1104 (1102 (1103))) like the above, respectively. About an input-medium name (1504) and an output media name (1506) (1507) as well as the above, with reference to a definition-part name of article (1107), a Japanese name is acquired and a definition-part name of article and a Japanese name are outputted from the input/output-media definition part (1106) of a program definition-part article, respectively. Whether it is an input medium among input/output media or it is an output media call an output media hereafter what is an input medium and "O" about that whose I/O partition is "I" among input/output media that what is necessary is just to refer to a corresponding I/O partition (1108).
[0086] About description of processing (1505) of functional description drawing, that in which input/output media carried out Japanese Natori profit is compounded and outputted to the description section (1302) of processing of a standard function recital article.
[0087] Next, the generation judging of a check condition table and a related check condition table is performed based on the standard-pattern name already acquired from the return program definition-part

article to drawing 3 (305). For example, the standard pattern "CHK01" currently used by this example is a pattern which reads an input file, checks the content and is written out to an output file. A standard pattern including such check processing of input data is judged with it being necessary to carry out the generation output of a check condition table and the related check condition table.
[0088] When judged with a check condition table and a related check condition table needing to be generated, a check condition table and a related check condition table are developed and outputted as follows.
[0089] First, the expansion approach of a check condition table is explained using drawing 20 .
[0090] The check condition table is developed about all the data items that were defined as the program definition-part article and that input define the file components and record definition components based on an input-medium definition-part name of article for every input medium, and are further used with the input medium, referring to a dictionary for every data item (306).
[0091] About the output method of a program name (1603) and a program symbolic name (1604), it is the same as that of said program manipulation schematic diagram. This input-medium definition-part name of article acquired from the program definition-part article and the Japanese Natori profit thing based on it are outputted to the definition-part name of article (1602) and its Japanese name (1601) of an input medium. The information about the output media of the output destination change of the error code edited by check processing or an error message is outputted to the definition-part name of article (1606) and its Japanese name (1605) of an output media. For this reason, among the output medias defined as the program definition-part article, an activity partition (1109) judges what is "ERR" to be the output destination change of an error message, and outputs that output-media definition-part name of article and the Japanese Natori profit thing based on it.
[0092] From the software components (701) of a dictionary (201), an attribute (1608) is acquired in the column of an input item from the definition-part article of an input medium, and this data item name and its Japanese name (1607) are outputted to it.
[0093] It outputs to the column of a keyword (1609), check conditions (1610), an error code (1611), and an error message (1612) based on the content defined as the check processing definition part (706) of a dictionary (201) by the column of check processing. Since the edit approach is described by the software components of a dictionary by the keyword about the check condition column, the semantics which a keyword shows is changed and outputted to Japanese.
[0094] In this example, first, since the input-medium definition-part name of article (1107) of a program definition-part article is "KYFIL", with reference to corresponding define the file components (drawing 14 (a)), "KYREC" which is a corresponding record definition components name (1010) is acquired. Furthermore, corresponding record definition components (drawing 14 (b)) are inputted, the input item column and the check processing column are developed in order of the data item defined, and it outputs and goes.
[0095] In the record layout definition (1012) of "KYREC", since the data item name (1014) defined as the 1st is "SIMEI-NO", with reference to the corresponding software components (drawing 7) of a dictionary (201), the "name number" which is the Japanese name (703) of a data item is acquired first. Next, from the check processing definition part (706) of a dictionary, a keyword (709), an error code (710), and an error message (711) are acquired and outputted. "SUB (SUB01)" of a keyword (1609) is a keyword which shows the software components of a subroutine call, and SUB01 expresses the program symbolic name of the subroutine called. So, the semantics which a keyword indicates "Checks by calling a subroutine (SUB01)" is developed and outputted to Japanese at the check condition column (1610) (the processing which changes into Japanese the semantics which such a keyword shows is henceforth called keyword Japanese conversion).
[0096] A check condition table is generated by repeating the same procedure to the data item name (1014) of a record layout definition (1012) below. Next, a return related check condition table is generated to drawing 3 (307). About all the input media defined as the program definition-part article, define the file components and record definition components are inputted based on an input-medium definition-part name of article, and the generation judging of a related check condition table is

performed for every data item about all the data items currently further used with the input medium.
[0097] Criteria generate the related check condition table to this related check definition-part article, only when all the associated data items (802) indicated by the related check definition-part article corresponding to that the related check definition-part article to this data item exists and this data item are included in the input medium used by this program. In addition, when there are two or more input media, the associated data item should just be included in one of the input media.
[0098] In this example, since "SYOKUI-CD" which a related check definition-part article exists about "SIKYU-GK" among the data item names (1014) in an input medium (drawing 12), and is indicated by the associated data subject name (802) of this related definition-part article is contained in the record definition components (drawing 14 (b)) to the input medium of this program, it judges with an input check condition table needing to be generated. Under the present circumstances, the input-medium definition-part name of article which becomes the input origin of an associated data item is acquired.
[0099] About data items other than "SIKYU-GK" in which a related check definition-part article does not exist, it judges not generating a related check condition table.
[0100] When judged with a related check condition table needing to be generated, the related check condition table is developed referring to this related check definition-part article and the software components of a dictionary (201).
[0101] The expansion approach of a related check condition table is explained using drawing 21 . About the output method of a program name (1701) and a program symbolic name (1702), it is the same as that of said program manipulation schematic diagram. About the symbolic name (1706) and its Japanese name (1705) on an input medium, the data item name acquired from the data item name (804) of a related check definition-part article and the Japanese Natori profit thing based on it are outputted. About conditions (1707) and a matrix (1708), it acquires from the conditions (805) and matrix (806) of a related check definition-part article, and outputs. About an error code (1711), an error message (1712), and a matrix (1713), it acquires from the error code (808), error message (809), and matrix (810) of a related check definition-part article, and outputs.
[0102] About the definition-part name of article (1704) and its Japanese name (1703) of an input medium, the definition-part name of article of the input medium with which this data item is included already acquired at the time of a related check generation judging, and the thing which performed Japanese Natori profit to it are outputted.
[0103] About the definition-part name of article (1710) and its Japanese name (1709) of an output media, it outputs completely like the error code in a check condition table, and an error message output destination change.
[0104] Next, the expansion approach of an edit condition table is explained using drawing 22 .
[0105] all the combination of the output media and input medium which were defined as the program definition-part article -- receiving -- the -- combining -- ** -- it is alike and define the file components, record definition components, and a document definition-part article are inputted based on this output-media definition-part name of article, and the edit condition table is developed about all the data items currently further used by the output media, referring to the software components of a dictionary for every data item (step 308 of drawing 3).
[0106] About the output method of a program name (1803) and a program symbolic name (1804), it is the same as that of said program manipulation schematic diagram. This output-media definition-part name of article and this input-medium definition-part name of article which were acquired from the program definition-part article, and the Japanese Natori profit thing based on it are outputted to the definition-part name of article (1802) of an output media and its Japanese name (1801), the definition-part name of article (1806) of an input medium, and its Japanese name (1805), respectively.
[0107] About the column of an output item, an output item name (1807) outputs a data item name and the Japanese Natori profit thing based on it, and an attribute (1808) acquires and outputs the attribute (912) defined on the output media to this data item name. A Japanese name (703) and an attribute (704) are acquired and outputted from the software components of a dictionary based on a data item name and it about the data item on the input medium which becomes the information origin at the time of editing

an output item about the input subject name (1812) and attribute (1813) of the input item column.
[0108] It outputs to the keyword column (1809), the edit approach column (1810), and the edit partition column (1811) based on the content defined as the input edit definition part (707) and output edit definition part (708) of a dictionary by the column of edit. Since the edit approach of the software components of a dictionary is described by the keyword about the edit approach column (1810), it outputs by performing keyword Japanese conversion. When the edit approach is developed from input edit of the software components of a dictionary by the edit partition column (1811) and it is developed from "input edit" and output edit by it, it outputs to it with "output edit."
[0109] The criteria at the time of developing the edit approach defined as the software components of the dictionary to a data item here to the keyword column (1809) and the edit approach column (1810) are explained.
[0110] First, the expansion criteria in input edit are shown. Although it is not included in the input medium of this program that this data item is newly generated based on the data in an input medium during that the input media to this program are media into which it is inputted by the user, such as an input screen or an input document, or this program execution, i.e., this data item, it is two points of being contained in an output media, and input edit will be developed if the method of either 1 is filled.
[0111] The expansion criteria in output edit are shown. The output medias of this program are media, such as an output screen or a document, and output edit is developed when this data item is included in both an input medium and an output media.
[0112] Hereafter, the example of generation of the edit condition table in this example is shown in a detail. Since the groups of the output media defined as the program definition-part article and an input medium are an output document "KYLST" (603), an input file "KYFIL" (601) and an output document "KYERR" (604), and an input file "KYFIL" (601), its attention is first paid to the 1st set of "KYLST(s)" and "KYFIL(s)." A document definition-part article (drawing 13 (a)) is inputted based on "KYLST", and its attention is paid to the data item defined there.
[0113] Since the data item defined as the 1st of a data item name (911) is "SIMEI-NO", from the software components (drawing 7) of a dictionary, a document definition-part article to an attribute "X (4)" (912) is acquired, and it outputs the Japanese name "a name number" (703) of a data item. Since "SIMEI-NO" is contained in an input medium, input edit is not developed. Next, nothing is defined although the output edit definition part of a dictionary is referred to. This shows that it outputs at the time of output edit, without also performing processing of what about this this data item. Then, it outputs, saying a keyword is made into a null and "is transmitted as it is" to the edit approach column (1810).
[0114] Since the data item defined as the 2nd of a data item name (911) is "SIMEI", similarly, from the software components (drawing 8) of a dictionary, a document definition-part name of article to an attribute "N (7)" (912) is acquired, and it outputs the Japanese name "a name" of a data item. Next, with reference to the input edit definition part (707) of a dictionary, the keyword "SUB (SUB02, SIMEI-NO)" of the edit approach is acquired. This is the keyword which shows the software components of a subroutine call, and it is shown that "SUB02" is an input parameter to which "SIMEI-NO" hands over a subroutine name to the subroutine. Therefore, it turns out that the input subject name (1812) to "SIMEI" is "SIMEI-NO."
[0115] Since "SIMEI" is not contained in "KYFIL" which is the input medium of this program, the expansion criteria in input edit are fulfilled. Therefore, an input item "SIMEI-NO" and its attribute are outputted to the column (1809 1810) of edit of what performed a keyword "SUB" and keyword Japanese transform processing at the input item column (1812 1813), respectively.
[0116] In addition, since nothing is defined as the output edit definition part of a dictionary in this example, only input edit is developed in this way, but when further defined also as the output edit definition part, both input edit and output edit are developed and outputted to this sequence.
[0117] Since the data item defined as the 3rd of a data item name (911) is "NYUSYA-YMD", it acquires the software components (drawing 9) of a dictionary to an "entrance date", and outputs a document definition-part name of article to an attribute "X (8)" (912) for it to the output item column (1807 1808),

respectively. Next, a keyword "DATE" is acquired with reference to the output edit definition part (708) of a dictionary. Since this output media is a document, the expansion criteria in output edit are fulfilled. Then, it outputs "it changing into the format of YY.MM.DD" to the keyword column (1809) and the edit approach column (1810), and "NYUSYA-YMD" and the attribute of self and a same name are acquired in the input item column (1812 1813) from the software components of a dictionary, and are outputted to it, respectively. [which performed keyword Japanese conversion with the keyword]

[0118] About the data item "SYOKUI-CD" defined as the 4th of a data item name (911), it develops like the case of "SIMEI-NO."

[0119] About the data item "SIKYU-GK" defined as the 5th of a data item name (911) Although there is no definition information in the input edit definition part of a dictionary, and an output edit definition part like "SIMEI-NO" The attributes acquired from the document definition-part article are "Z, ZZZ, ZZ9" (912), and since this performs zero suppress and comma edit, it outputs to the edit approach column (1810) in this case, saying "It edits into the format of Z, ZZZ, and ZZ9."

[0120] Hereafter, the procedure of program generation is explained.

[0121] In advance of explanation of the procedure of program generation, the storing format of the program specification on a program specification library (210) is explained using drawing 23 - drawing 26 .

[0122] Drawing 23 is the storing format of the program manipulation schematic diagram on a program specification library (210). This format is constituted by the column of the program name (1901) of a user program, a symbolic name (1902), the standard-pattern name (1903) used by the user program, the Japanese name (1904) of input/output media, a definition-part name of article (1905), an I/O partition (1906), an activity partition (1907), a logical unit name (1908), a prefix (1909), and the processing outline (1910) of a program. This is constituted by the completely same data as the example of an output of the program manipulation schematic diagram in drawing 18 .

[0123] Drawing 24 is the storing format of the check condition table on a program specification library (210). This format is constituted by the column of the definition-part name of article (2001) of an input medium, the definition-part name of article (2002) of an output media, the Japanese name (2003) of an input item, a symbolic name (2004), an attribute (2005), the keyword (2006) of check processing, check conditions (2007), an error code (2008), and an error message (2009). This is constituted by the completely same data as the example of an output of the check condition table in drawing 20 .

[0124] Drawing 25 is the storing format of the related check condition table on a program specification library (210). This format is constituted by the column of the keyword (2104) of the definition-part name of article (2101) of input/output media, the Japanese name (2102) of an input item, a symbolic name (2103), and check conditions, an error code (2105), an error message (2106), and a matrix (2107). This is constituted by the completely same data as the example of an output of the related check condition table in drawing 21 .

[0125] Drawing 26 is the storing format of the edit condition table on a program specification library (210). This format is constituted by the column of the keyword (2206) of the definition-part name of article (2201) of an input medium, the definition-part name of article (2202) of an output media, the Japanese name (2203) of an output item, a symbolic name (2204), an attribute (2205), and the edit approach, the edit approach (2207), an edit partition (2208), the Japanese name (2209) of an input item, a symbolic name (2210), and an attribute (2211). This is constituted by the completely same data as the example of an output of the edit condition table in drawing 22 .

[0126] Next, the procedure of program generation is explained based on a flow chart.

[0127] Drawing 4 is a flow chart which shows the procedure of program generation, and drawing 27 - drawing 29 are the examples of expansion of the source program by the program generation function based on the definition-part article (drawing 13 - drawing 14) in program specification (drawing 23 - drawing 26) and a data definition document. Hereafter, the concrete procedure of a program generation function is explained using this example.

[0128] A generation function acquires the standard-pattern name "CHK01" (1903) first used from the program manipulation schematic diagram (1900) of the user program for generation, and a

corresponding standard pattern (1202) is inputted from a standard-pattern library (205) (step 401).
[0129] The definition-part article of input/output media is inputted from a data definition document (203) based on the input/output-media name which the I/O bill of materials (402) defined with the program manipulation schematic diagram of this program, the source code of the I/O declaratives of this program is generated, and it compounds to the standard pattern which inputted this at step 401.
[0130] The expansion approach of an I/O declaratives is concretely explained using drawing 27 . A program generation function from the input/output-media definition-part name of article (1905) indicated by the program manipulation schematic diagram (1900) "KYFIL", Acquire definition-part names of articles, such as "KYLST", and define the file components (1001) and a document definition-part article (901 907) are read. [/ based on this] An I/O environmental definition (2302 2303), the file description (2304 2306), and the data description (2305 2307) of this program are generated from such information, and this is compounded to a standard pattern (1202).
[0131] Document printing processing expansion (404) is compounded to the standard pattern which carried out when this program was using the standard pattern which outputs a document, inputted the document definition-part article which corresponds from a data definition document (203) based on the output-media definition-part name of article indicated by the program manipulation schematic diagram of this program, from now on, generated the working area for document edit, and the document output procedure, and was inputted at step 401.
[0132] Document printing processing expansion processing (404) is concretely explained using drawing 27 . A program generation function reads a document definition-part article (901 907) based on the document definition-part name of article (1905) indicated by the program manipulation schematic diagram (1900), generates working areas (2308), such as a counter used from these contents of a definition at the time of the working area (2309) for document edit about a document, and program execution, and compounds this to a standard pattern (1202).
[0133] In a check, a related check, and edit processing expansion (405), based on the keyword indicated by the check condition table of the program specification, the related check condition table, and the edit condition table, processing based on data items is generated and this is developed, respectively to the check processing section (1203) of a standard pattern, the related check processing section (1204), and the edit processing section (1205).
[0134] The source program expansion approach is concretely explained taking the case of the expansion from an edit condition table using drawing 29 . The edit condition table (2200) of this program is read first, the processing to this data item is generated based on the edit conditions defined as this edit condition table by the keyword (2206), and this is developed in the edit processing section (1205) of a standard pattern, respectively. It generates simultaneously also about a working area required for a data item editing in that case. For example, in the case of the keyword "DATE" (2206) of the 3rd line of an edit condition table (2200), as shown in a field (2310), the working area for dividing the date into a year, the moon, and a day and the working area for edit are generated.
[0135] In case the processing to a data item is furthermore developed, the prefix (1909) which the program manipulation schematic diagram (1900) defined as this example showed is added to the subject name on each file. For example, as shown in a statement (2314), to the data item "SIKYU-GK", "IN -" was added to the input item, "OU -" was added to the output item, respectively, and the notation subject name on an input file has determined the notation subject name on "IN-SIKYU-GK" (2315) and an output file as "OU-SIKYU-GK" (2316) and a program generate time. The software components based on data items are made to become independent of the definition-part article of input/output media, such as a file treating that data item, by this.
[0136] In addition, drawing 28 replaces the file symbolic name of the undefined in a standard pattern (1202) by the file symbolic name (1004) of define the file components.
[0137] With a source program output (406), the user program of the object currently generated by the above processing is outputted to a source program library (211).
[0138] In addition, in this example, although the example of the program of batch processing was given and explained, it is generable similarly about the program which performs access to the program of

interactive processing accompanied by access to a database, the resource on other equipments, or a process.

[0139] Moreover, in this example, although the source program is generated from program specification, it is also possible to input a dictionary (201), a related check definition document (202), a data definition document (203), a program definition document (204), and a standard function explanation library (206), without minding program specification, and to generate a direct source program, and also when using this approach, it is contained in the range of this invention. In this case, it is the same point as a program specification generate time, and the software components based on data items are chosen based on the data item which exists in the input/output media of the target user program, and it compounds to a standard pattern.

[0140] It doubles and the program specification edit function (212) and software components reverse generation function (208) in this example are explained to the last, while the example of an activity of these functions is shown. The example of an activity to be introduced from now on is an example which makes program specification correction using a program specification edit function (212), and corrects software components from the program specification after correction to the example program shown in drawing 6 using a software components reverse generation function (208).

[0141] It introduces about the example of correction of various program specification first. Drawing 24 (a) is the example of the check condition table immediately after generation, and drawing 24 (b) is the example of the check condition table after correction. Drawing 25 (a) is the example of the related check condition table immediately after generation, and drawing 25 (b) is the example of the related check condition table after correction. Drawing 26 (a) is the example of the edit condition table immediately after generation, and drawing 26 (b) is the example of the edit condition table after correction.

[0142] Here, an edit condition table (drawing 26) is taken up and the content of correction is introduced to a detail. In the 3rd line (2222) of drawing 26 (b), the edit approach was changed [at it] into the null with the keyword "it transmitting as it is" from "format of YY.MM.DD --" from "DATE" (this actuation is considered as correction 1). This means not performing processing of what to data, either to a data item "NYUSYA-YMD" at the time of output edit. Moreover, in drawing 26 (b), the 4th line (2212) in drawing 26 (a) was deleted (it considers as this correction 2). This means deleting a data item "SYOKUI-CD" from an output media. In the 4th line (2223) of drawing 26 (b), the attribute of an output item "SIKYU-GK" was changed into "Z, ZZ9" from "Z, ZZZ, ZZ9." Furthermore, it was presupposed from the null the edit approach "is edited into the format of Z and ZZ9 per 1000 yen" with a keyword at it at "COMP (SIKYU-GK/1000)" (it considers as this correction 3). In case this outputs "SIKYU-GK", it newly defines output edit processing in which the value divided by 1000 is outputted.

[0143] In addition, about correction of the program specification in a program specification library (210), it can also carry out using the general-purpose software for spreadsheets, or the software for document preparation, without using an exclusive editor, since it stores in the image of a simple table as program specification is shown in drawing 23 - drawing 26 , This is also contained in the range of this invention. Furthermore, it transmits on the equipment group B (118) which is equipment different from the equipment group A (108) which generated program specification, and it is also possible on the equipment group B to perform edit using the general-purpose software for spreadsheets or the software for document preparation, and this is also contained in the range of this invention.

[0144] Next, the procedure of software components reverse generation is explained using drawing 5 . The reverse generation said by this example means rewriting the software components which are the information origin on this based on the program specification with which correction was added, and making the content of correction of program specification reflect in it.

[0145] In the software components reverse generation function (208) of this example, a program manipulation processing schematic diagram is inputted (501), and reverse generation is carried out for a program definition document (204) (502). Next, a check condition table is inputted (503) and reverse generation is performed for a dictionary (201) and a data definition document (203) (504). Next, a related check condition table is inputted (505) and reverse generation is performed for a dictionary (201), a related check definition document (202), and a data definition document (203) (506). Next, an

edit condition table is inputted (507) and reverse generation is performed for a dictionary (201) and a data definition document (203) (508).
[0146] Next, a reverse generation result is explained. When reverse generation is performed based on the program specification after the above-mentioned correction, only what had modification in the content of a definition a reverse generation front and after reverse generation is shown in drawing 30 - drawing 36 . Drawing 30 - drawing 34 are the examples of the software components of the dictionary after reverse generation, and the examples of a definition before reverse generation are drawing 7 - drawing 11 . Drawing 35 is the example of the related check definition-part article after reverse generation, and the example of a definition before reverse generation is drawing 12 . Drawing 36 is the example of the document definition-part article to "KYLST" among the data definition documents after reverse generation, and the example of a definition before reverse generation of this is drawing 13 (a).
[0147] The concrete procedure of reverse generation is explained based on the example of an edit condition table [finishing / the above-mentioned correction] (drawing 26 (b)).
[0148] The keyword (2214) is changed into the null in correction 1. In this case, first, with reference to an edit partition (2215), since this is "output edit", a reverse generation function acquires an input item "NYUSYA-YMD" (2216) with reference to a symbolic name (2210). The software components of the dictionary (201) corresponding to it are set as the object of reverse generation. The keyword is changed into the null as shown in the input edit definition part (2401) of the software components (drawing 32) of the dictionary after reverse generation.
[0149] In drawing 26 (b), the data item "SYOKUI-CD" (2212) was deleted from the output item by correction 2. In this case, a reverse generation function deletes "SYOKUI-CD" from the document definition-part article "KYLST" (drawing 13 (a)) of the document which is the output media of this program. In the document definition-part article after reverse generation (drawing 36), the data item "SYOKUI-CD" is deleted as shown in the document layout definition part (2601).
[0150] In correction 3, the keyword is changed like correction 1. Also in this case, since an edit partition is "output edit" (2220), it acquires a symbolic name "SIKYU-GK" (2221), and the output edit definition part of the software components (drawing 34) of the dictionary corresponding to it is set as the object of reverse generation. Since the keyword (2206) after correction is "COMP (SIKYU-GK/1000)" (2219), it updates the keyword (2402) of output edit of the software components of the dictionary to "SIKYU-GK."
[0151] Moreover, in correction 3, the attribute (2218) of a symbolic name "SIKYU-GK" (2217) is changed further. In this case, this data item attribute in the document definition-part article "KYLST" (drawing 13 (a)) of the document which is an output media is changed. The content of modification is reflected as shown in (2602) of the document layout definition part of the document definition-part article after reverse generation (drawing 36).
[0152] Thus, with the procedure of a program specification generation function (209), a software components reverse generation function (208) specifies software components with correction from a program specification library (210) in the procedure of reverse, and updates software components based on the content of the program specification library (210). Reverse generation of software components can be performed like an edit condition table also about the case where it carries out based on a program manipulation schematic diagram, a check condition table, and a related check condition table.

[Translation done.]

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]
[Drawing 1] It is hardware configuration drawing of the system for realizing the example of this invention.
[Drawing 2] It is the functional block diagram of the system which realizes this invention.
[Drawing 3] It is the flow chart which shows the procedure of program specification generation.
[Drawing 4] It is the flow chart which shows the procedure of program generation.
[Drawing 5] It is the flow chart which shows the procedure of software components reverse generation.
[Drawing 6] It is drawing showing the example of a user program used for explanation of this example.
[Drawing 7] It is drawing showing the example of a definition of a dictionary.
[Drawing 8] It is drawing showing the example of a definition of a dictionary.
[Drawing 9] It is drawing showing the example of a definition of a dictionary.
[Drawing 10] It is drawing showing the example of a definition of a dictionary.
[Drawing 11] It is drawing showing the example of a definition of a dictionary.
[Drawing 12] It is drawing showing the example of a definition of a related check processing definition-part article.
[Drawing 13] It is drawing showing the example of a definition of a document definition-part article.
[Drawing 14] It is drawing showing the example of a definition of define the file components and record definition components.
[Drawing 15] It is drawing showing the example of a definition of a program definition-part article.
[Drawing 16] It is drawing showing the example of standard-pattern components.
[Drawing 17] It is drawing showing the example of a standard function recital article.
[Drawing 18] It is drawing showing the example of an output of a program manipulation schematic diagram.
[Drawing 19] It is drawing showing the example of an output of a program function explanatory view.
[Drawing 20] It is drawing showing the example of an output of a check condition table.
[Drawing 21] It is drawing showing the example of an output of a related check condition table.
[Drawing 22] It is drawing showing the example of an output of an edit condition table.
[Drawing 23] It is drawing showing the example of the storing format of a program manipulation schematic diagram.
[Drawing 24] It is drawing showing the example of the storing format of a check condition table.
[Drawing 25] It is drawing showing the example of the storing format of a related check condition table.

 [Drawing 26] It is drawing showing the example of the storing format of an edit condition table.
[Drawing 27] It is drawing showing the example of source program expansion by the program generation function.
[Drawing 28] It is drawing showing the example of source program expansion by the program generation function.
[Drawing 29] It is drawing showing the example of source program expansion by the program generation function.

[Drawing 30] It is drawing showing the example of a definition of the dictionary after reverse generation.
[Drawing 31] It is drawing showing the example of a definition of the dictionary after reverse generation.
[Drawing 32] It is drawing showing the example of a definition of the dictionary after reverse generation.
[Drawing 33] It is drawing showing the example of a definition of the dictionary after reverse generation.
[Drawing 34] It is drawing showing the example of a definition of the dictionary after reverse generation.
[Drawing 35] It is drawing showing the example of a definition of the related check definition-part article after reverse generation.
[Drawing 36] It is drawing showing the example of a definition of the document definition-part article after reverse generation.
[Description of Notations]
201 -- A dictionary, 202 -- A related check definition document, 203 -- Data definition document, 204 -- A program definition document, 205 -- A standard-pattern library, 206 -- Standard function explanation library, 207 -- A program specification generation function, 208 -- Software components reverse generation function, 209 -- A program generation function, 210 -- Program specification library, 211 -- A source program library, 212 -- Program specification edit function, 213 -- Program specification, 701 -- A data item definition part, 702 -- Data item name, 703 [-- Check processing definition part,] -- A Japanese name, 704 -- An attribute, 705 -- A comment, 706 707 -- An input edit definition part, 708 -- An output edit definition part, 709 -- Check conditions, 710 -- An error code, 711 -- An error message, 802 -- Associated data subject name, 803 -- The condition definition section, 807 -- An error-processing definition part, 901 -- Document attribute definition part, 902 [-- Document layout definition part,] -- A document definition-part name of article, 903 -- A document name, 904 -- A document symbolic name, 907 1001 -- A file attribute definition part, 1002 -- A define the file components name, 1003 -- File name, 1004 -- A file symbolic name, 1005 -- A record definition components name, 1009 -- Record attribute definition part, 1010 -- A record definition components name, 1011 -- A record name, 1012 -- Record layout definition part, 1101 -- A program attribute definition part, 1102 -- A symbolic name, 1103 -- Program name, 1104 -- A standard-pattern name, 1105 -- A processing outline, 1106 -- Input/output-media definition part, 1107 -- A definition-part name of article, 1108 -- An I/O (I/O) partition, 1109 -- Activity partition, 1110 [-- A standard pattern, 1203 / -- The check processing section, 1204 / -- The related check processing section 1205 / -- The edit processing section, 1301 / -- A standard function recital name of article, 1302 / -- Standard function recital article] -- A logical unit name, 1111 -- A prefix, 1201 -- A standard-pattern name, 1202

[Translation done.]

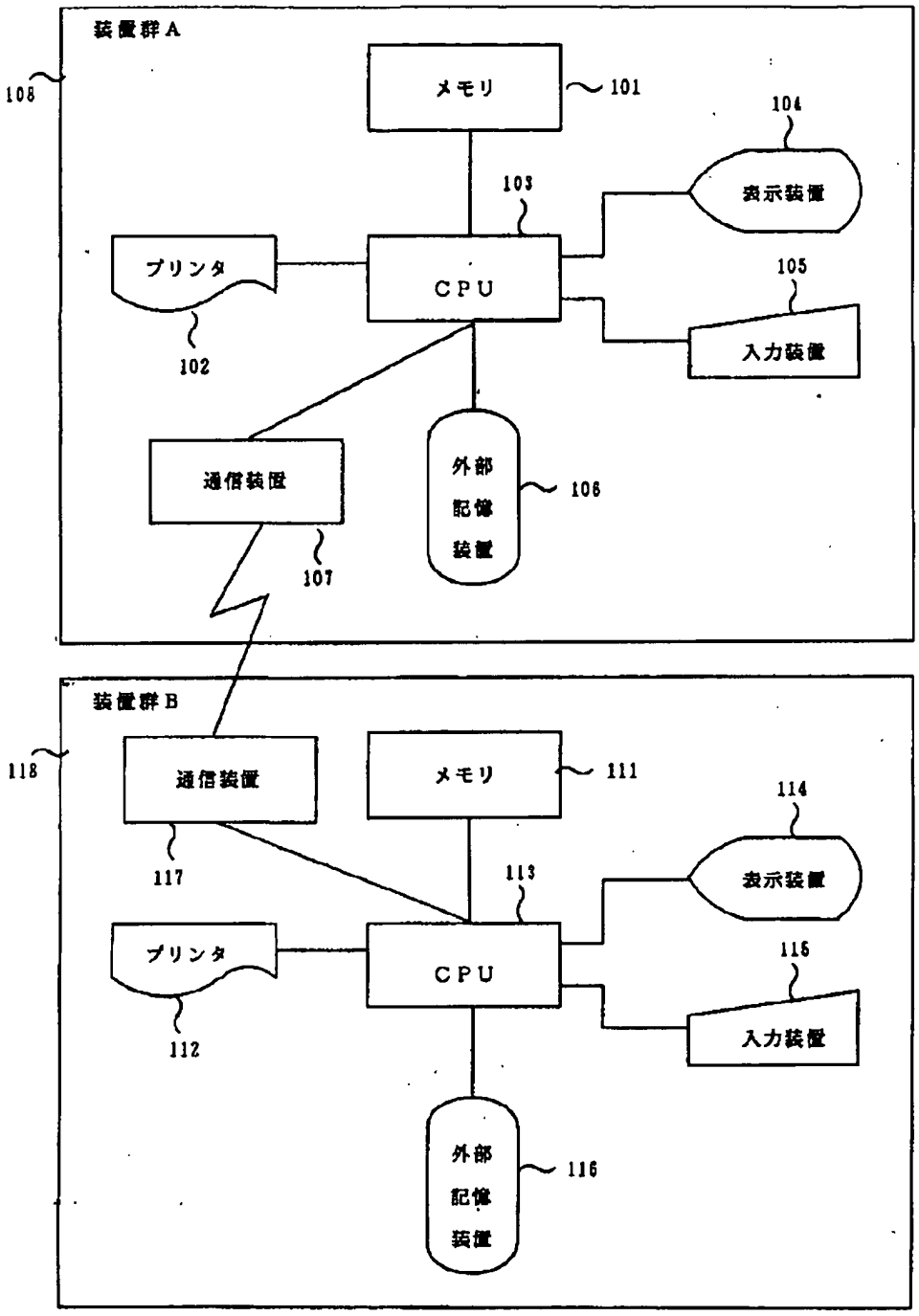
* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DRAWINGS

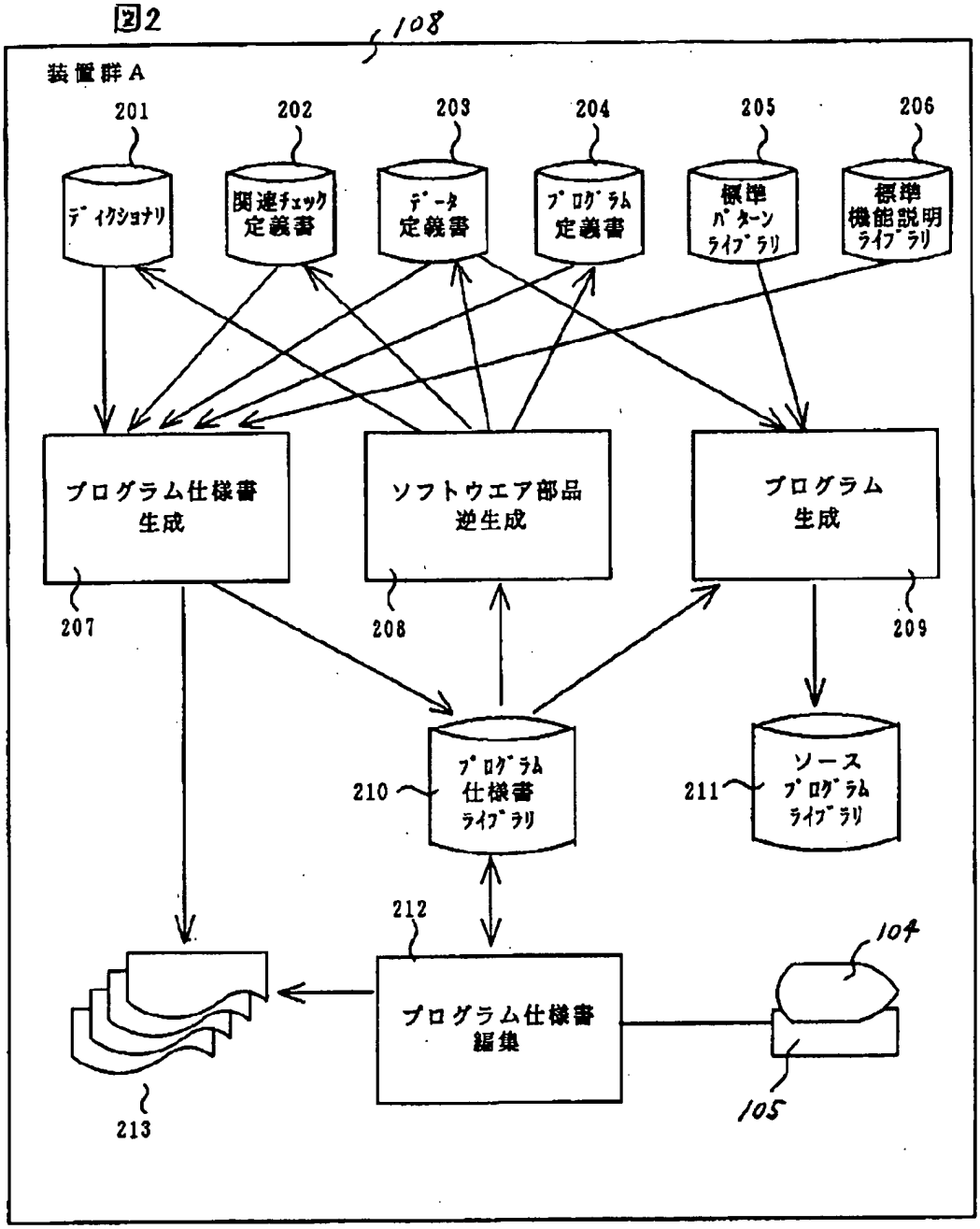
[Drawing 1]

図1


装置群A
108
メモリ
101
104
表示装置
103
プリンタ
CPU
105
入力装置
102
通信装置
外部
記憶
装置
106
107
装置群B
118
通信装置
メモリ
111
114
表示装置
117
113
プリンタ
CPU
116
入力装置
112
外部
記憶
装置
116

[Drawing 2]

図2


108
装置群A
201
202
203
204
205
206
ディクショナリ
関連チェック定義書
データ定義書
プログラム定義書
標準パターンライブラリ
標準機能説明ライブラリ
プログラム仕様書生成
ソフトウエア部品逆生成
プログラム生成
207
208
209
プログラム仕様書ライブラリ
ソースプログラムライブラリ
210
211
212
プログラム仕様書編集
104
105
213

[Drawing 3]

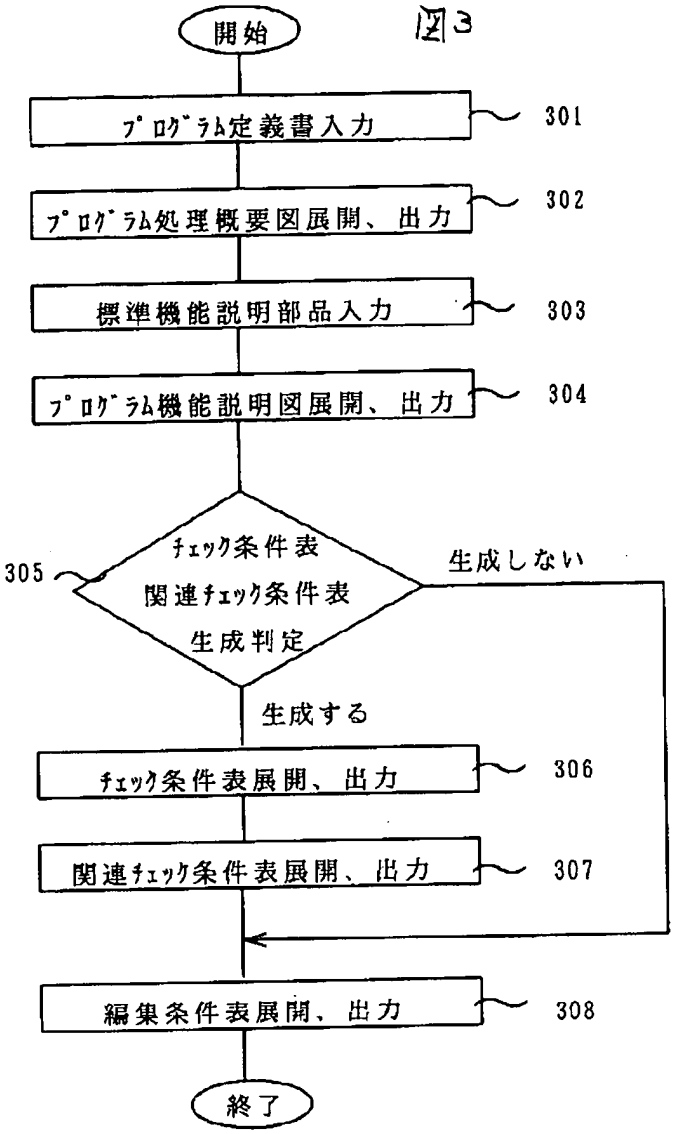
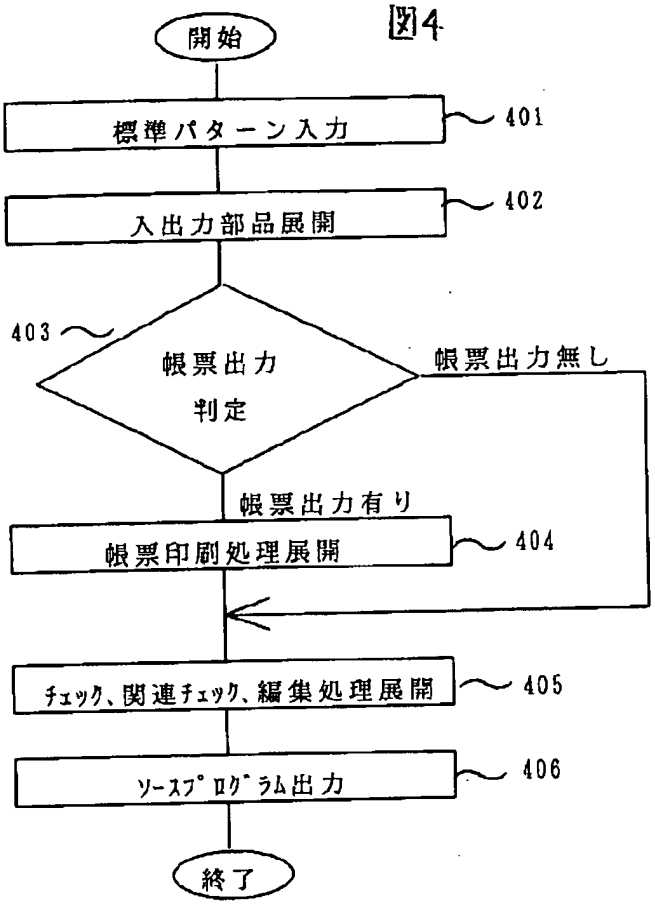


図3
開始
プログラム定義書入力
301
プログラム処理概要図展開、出力
302
標準機能説明部品入力
303
プログラム機能説明図展開、出力
304
305
チェック条件表
関連チェック条件表
生成判定
生成しない
生成する
チェック条件表展開、出力
306
関連チェック条件表展開、出力
307
編集条件表展開、出力
308
終了

[Drawing 4]

図4



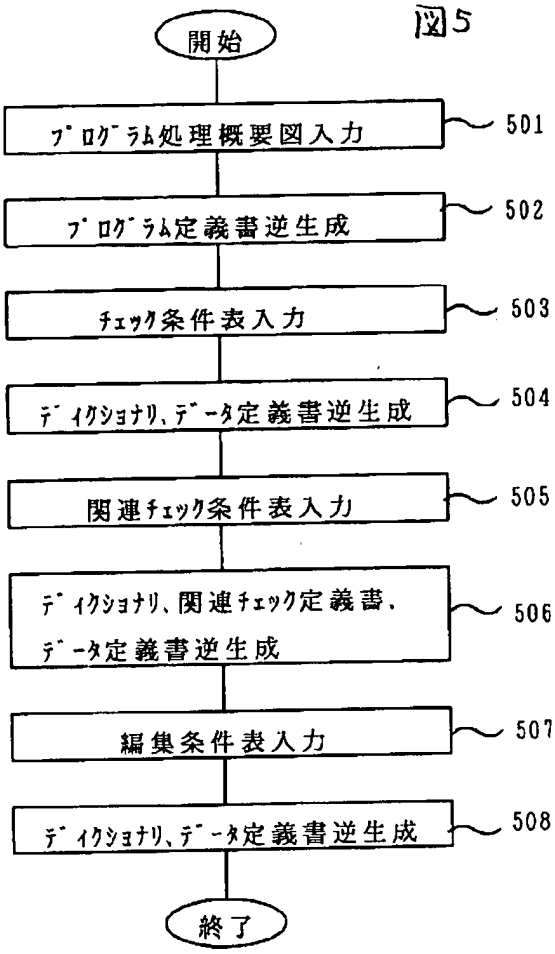
[Drawing 28]

図28

```
PROCEDURE              DIVISION.                              ~1202
MAIN-PROC              SECTION.
MAIN-010.
    OPEN  KYUYO-FILE KYUYO-LIST ERROR-LIST.
    PERFORM  KYUYO-FILE-READ-PROC.
    PERFORM  WITH TEST BEFORE
          UNTIL  KYUYO-FILE-END-SW = 'END'
      MOVE  SPACE    TO  ERR-CODE
      PEROFORM  UNT-CHK-PROC
      IF  ERR-CODE  =  SPACE
        THEN
          PERFORM  REL-CHK-PROC
        ELSE
          CONTINUE
      END-IF
      IF  ERR-CODE  =  SPACE
        THEN
          PERFORM  NOERR-OUTPUT-EDIT-PROC
          PERFORM  KYUYO-LIST-WRITE-PROC
        ELSE
          CONTINUE
      END-IF
      PERFORM  KYUYO-FILE-READ-PROC
    END-PERFORM.
    CLOSE  KYUYO-FILE KYUYO-LIST ERROR-LIST.
MAIN-999.
    STOP RUN.
          :
```

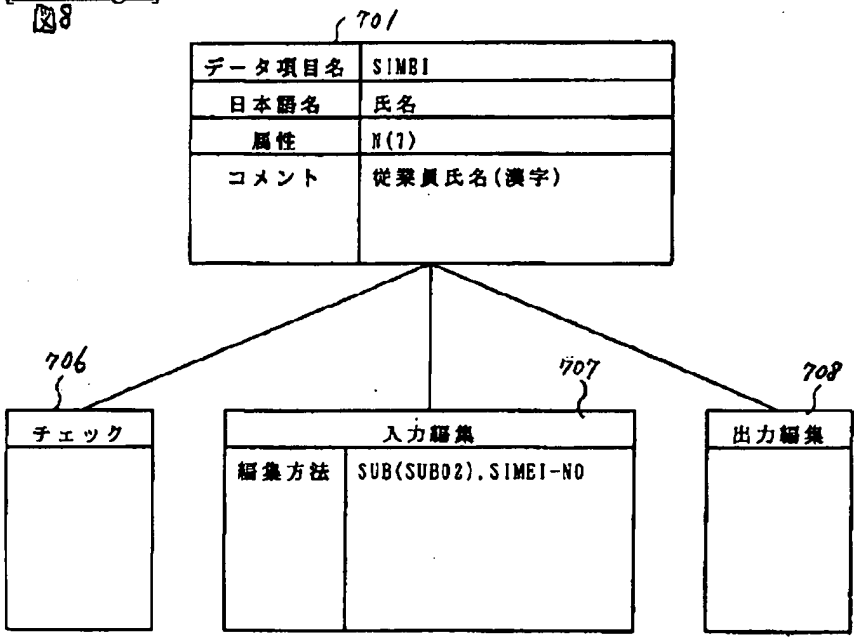
[Drawing 5]

図5



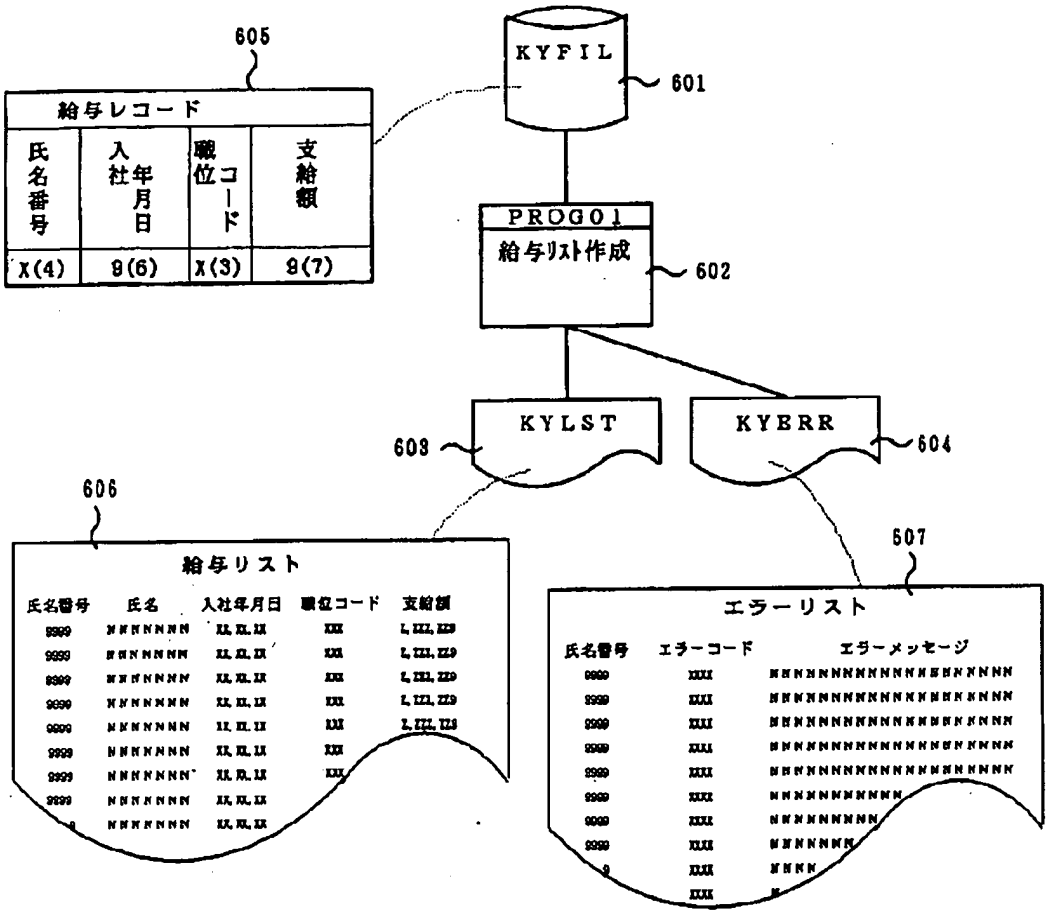
[Drawing 8]

図8



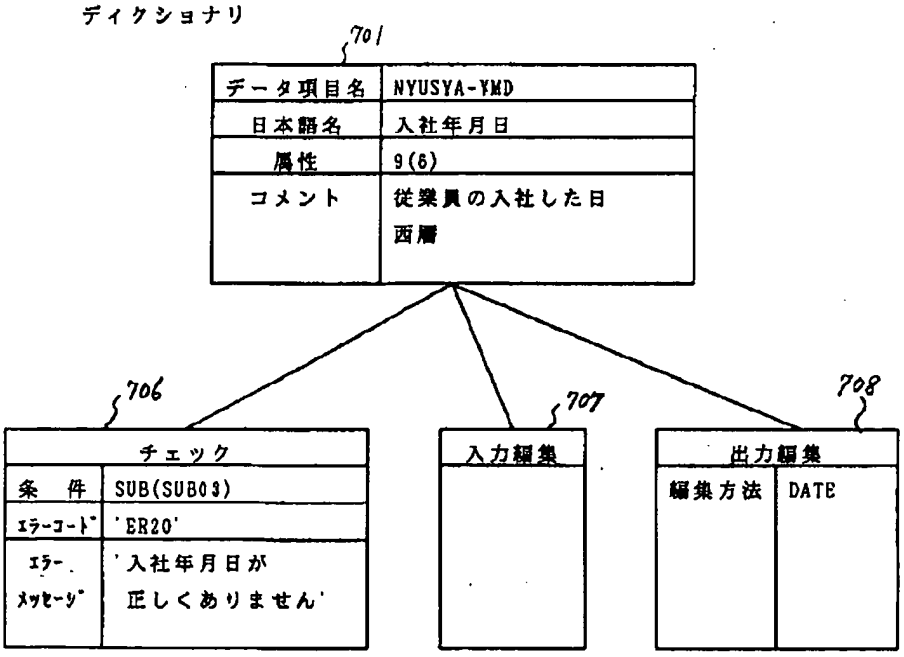
[Drawing 6]

図6



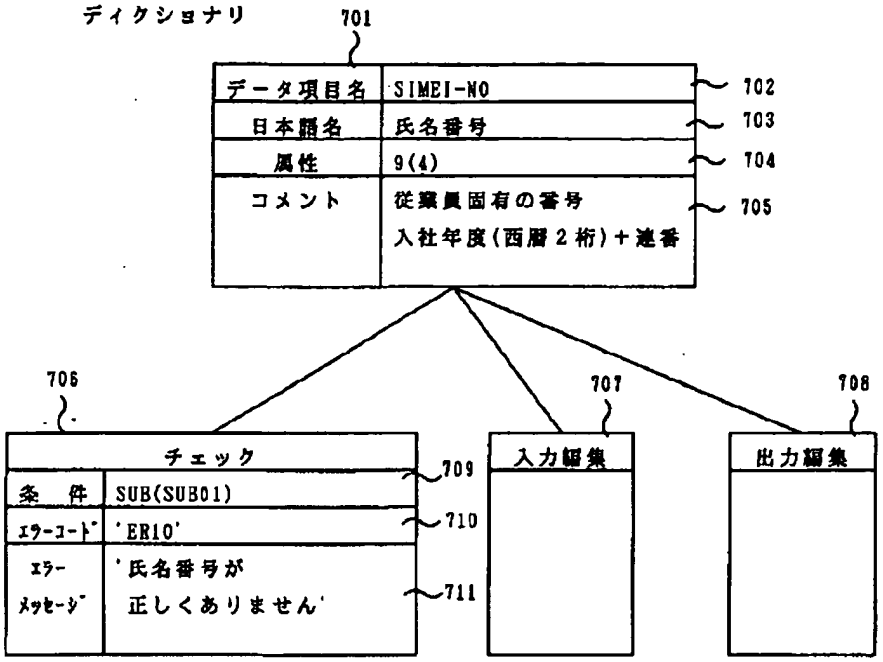
[Drawing 9]

図9



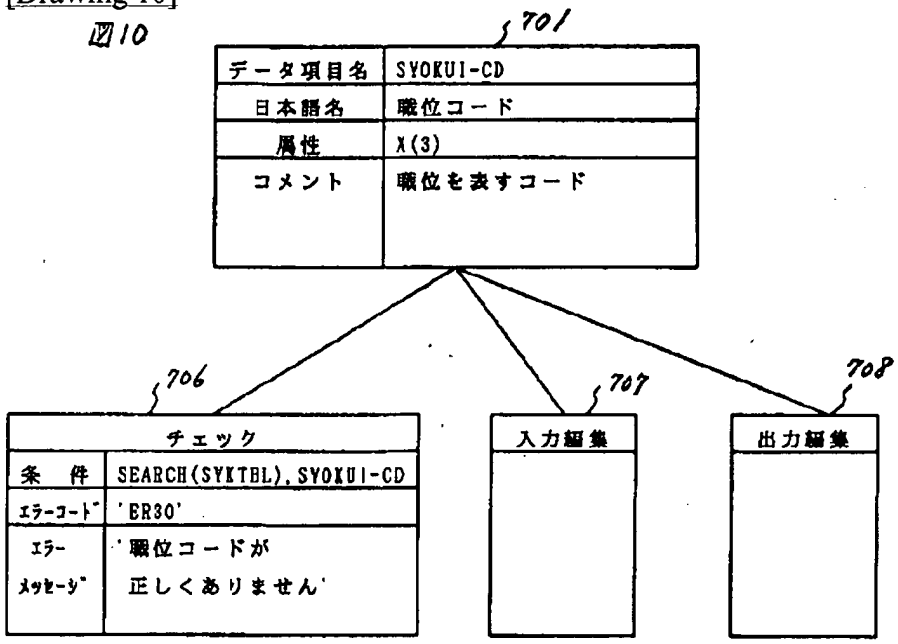
[Drawing 7]

図7



[Drawing 10]

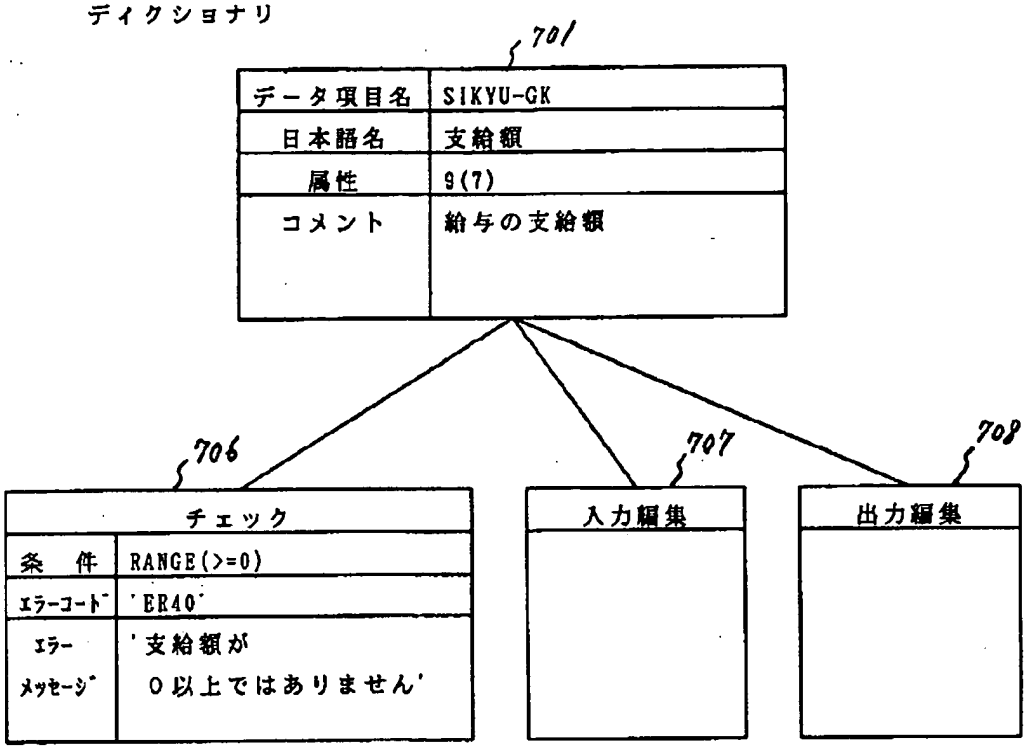
図10



[Drawing 11]

図11

ディクショナリ



[Drawing 15]

図15

プログラム定義部品

	1102 記号名	1103 プログラム名	1104 標準パターン名	1105 処理概要
1101～	PROG01	給与リスト作成	CHK01	給与ファイルを読み込み、データのチェックを行い、給与リスト及びエラーリストを作成する。

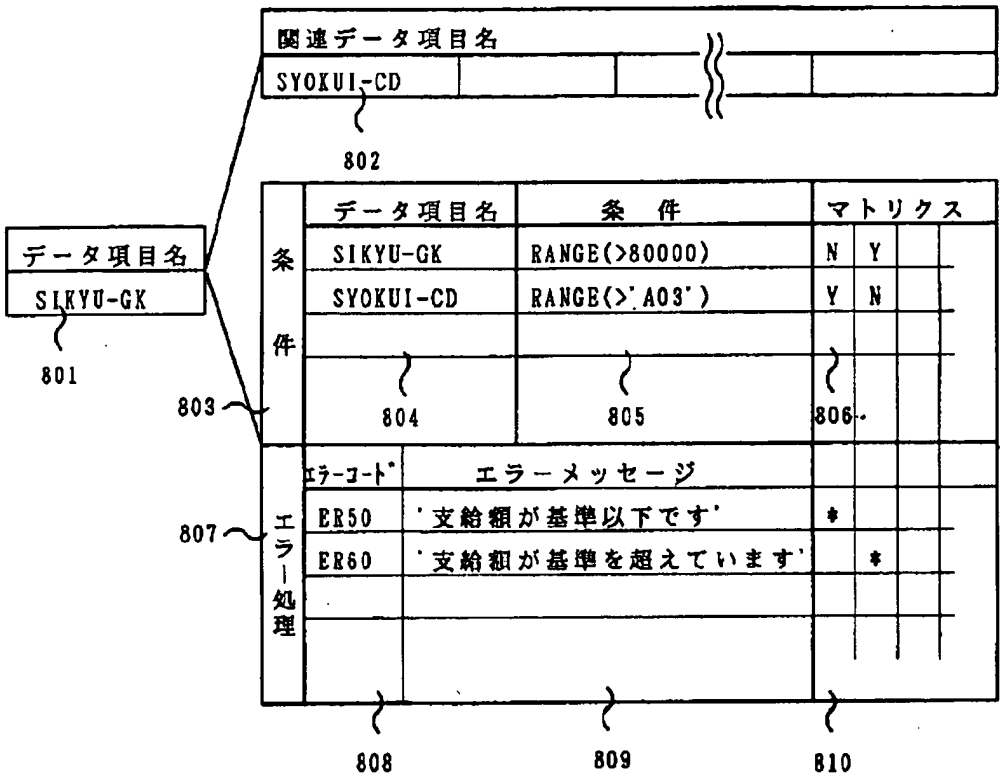
1106～ 入出力媒体定義

1107 定義部品名	1108 I/O区分	1109 使用区分	1110 論理装置名	1111 接頭語
KYFIL	I		SYS010	IN-
KYLST	O		SYS110	OU-
KYERR	O	ERR	SYSERR	ER-

[Drawing 12]

図12

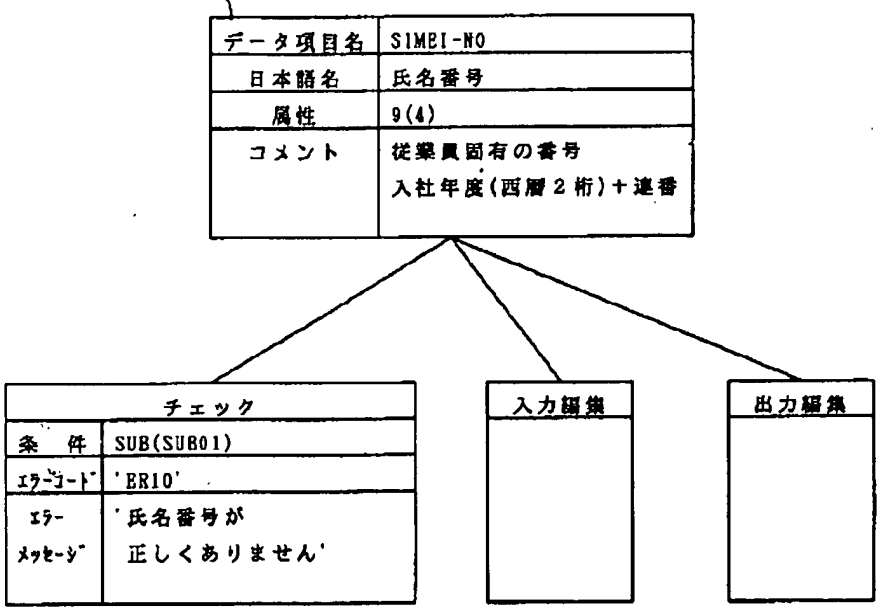
関連チェック定義部品



[Drawing 30]

ディクショナリ（逆生成後）

図30



[Drawing 13]

図13

(a) 帳票定義部品

902	903	904	905	906
帳票定義部品名	帳票名	帳票記号名	ﾚｺｰﾄﾞ長	印刷行数
KYLST	給与リスト	KYUYO-LIST	255	50

901～

907～ 帳票レイアウト定義

開始行	開始列	反復数	ﾃﾞｰﾀ項目名	属性
4	3	20	SIMEI-NO	X(4)
4	10	20	SIMEI	N(7)
4	50	20	NYUSYA-YMD	X(8)
4	60	20	SYOKUI-CD	X(3)
4	65	20	SIKYU-GK	ZZZ,ZZZ,ZZ9

908 909 910 911 912

(b) 帳票定義部品

901～

帳票定義部品名	帳票名	帳票記号名	ﾚｺｰﾄﾞ長	印刷行数
KYERR	エラーリスト	ERROR-LIST	80	66

907～ 帳票レイアウト定義

開始行	開始列	反復数	ﾃﾞｰﾀ項目名	属性
3	3	50	SIMEI-NO	X(3)
3	7	50	ERR-CODE	X(4)
3	12	50	ERR-MSG	N(20)

[Drawing 31]

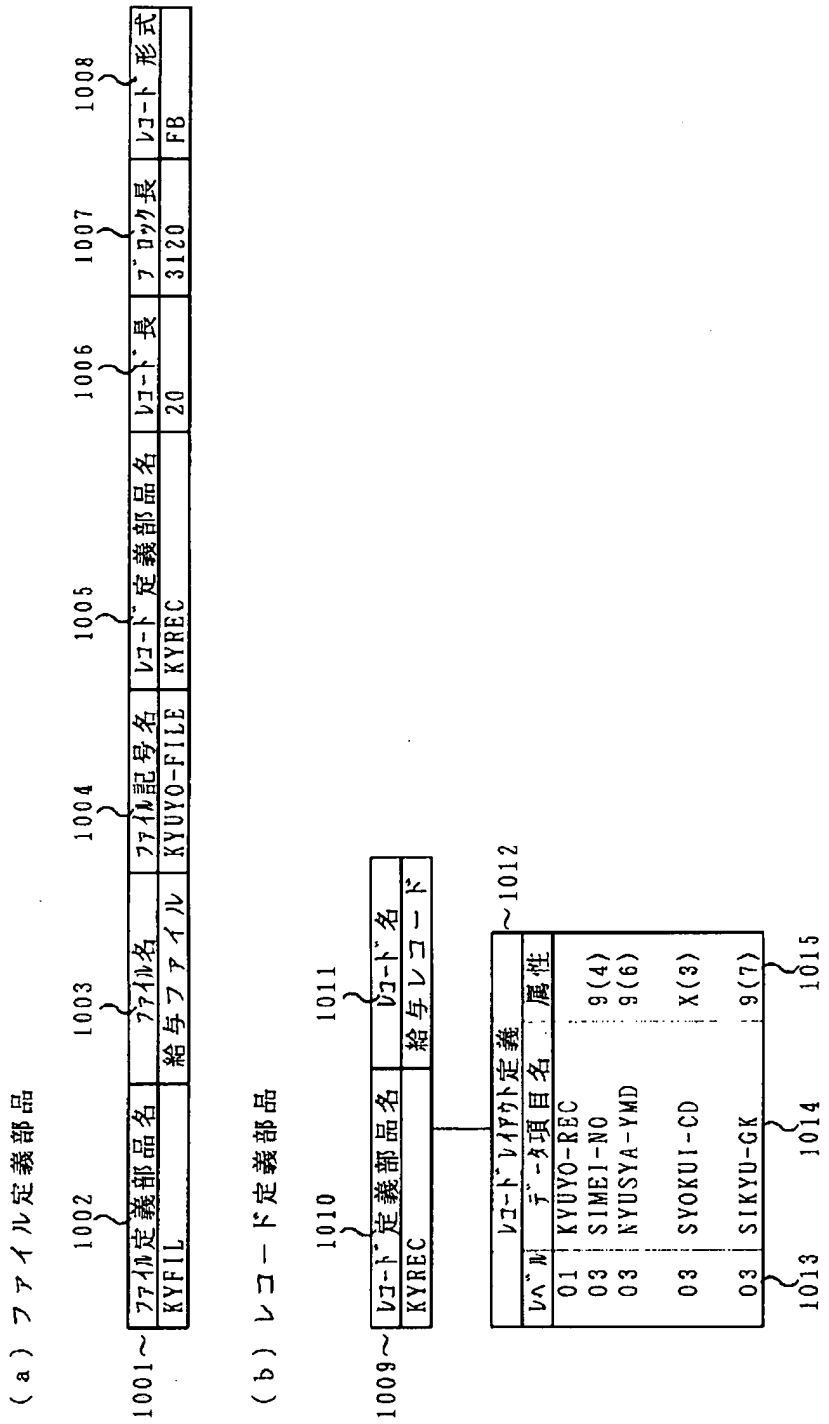
図31

データ項目名	SIMEI
日本語名	氏名
属性	N(7)
コメント	従業員氏名（漢字）

チェック	入力編集		出力編集
	編集方法	SUB(SUB02),SIMEI-NO	

[Drawing 14]

図14



[Drawing 23]

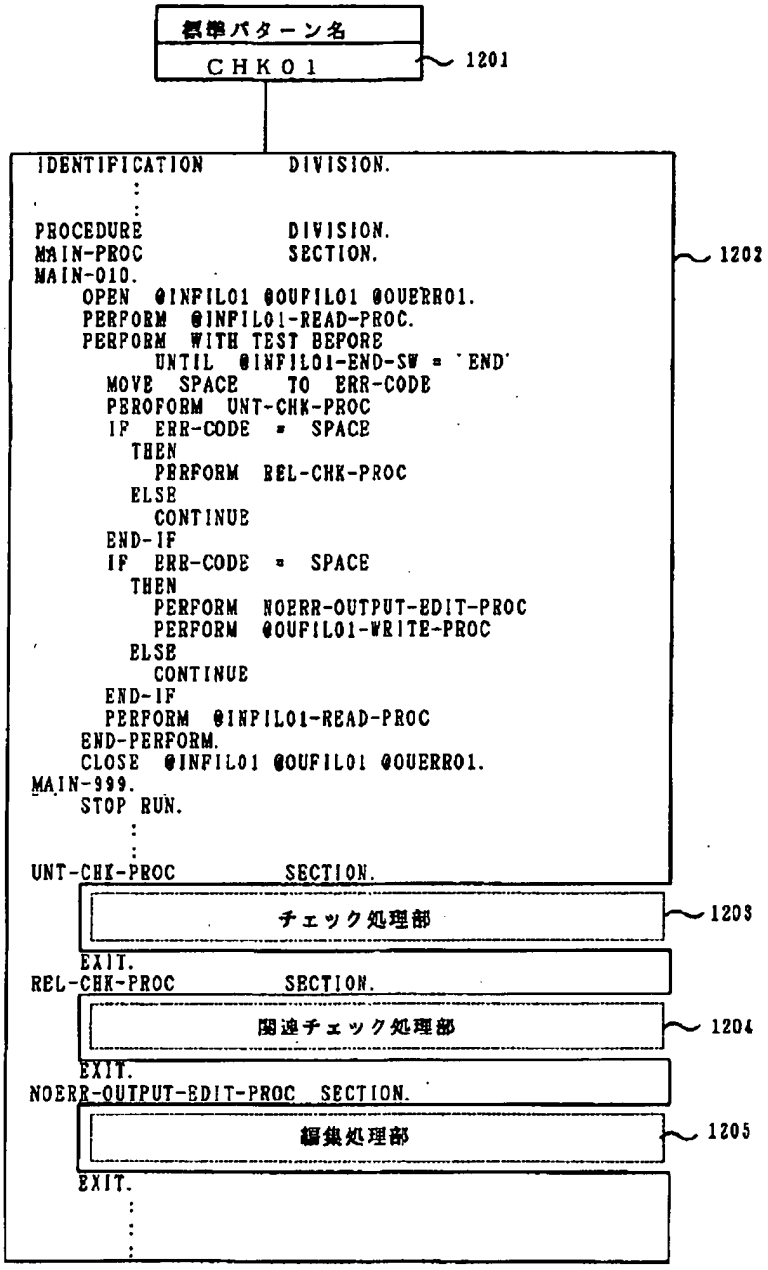
図23

1900～

順番	プログラム名 (1901)	記号名 (1902)	展開パターン名 (1903)	入出力媒体名 (1904)	定義部品名 (1905)	I/O区分 (1906)	使用区分 (1907)	論理装置名 (1908)	論理順 (1909)	処理概要 (1910)
1	給与リスト作成	PROG01	CHI01	給与ファイル	KYFIL	I		SYS010	IN-	給与ファイルを読み込み…
2	給与リスト作成	PROG01	CHI01	給与リスト	KYLST	O		SYS110	OU-	給与ファイルを読み込み…
3	給与リスト作成	PROG01	CHI01	エラーリスト	KYERR	O	ERR	SYSERR	ER-	給与ファイルを読み込み…

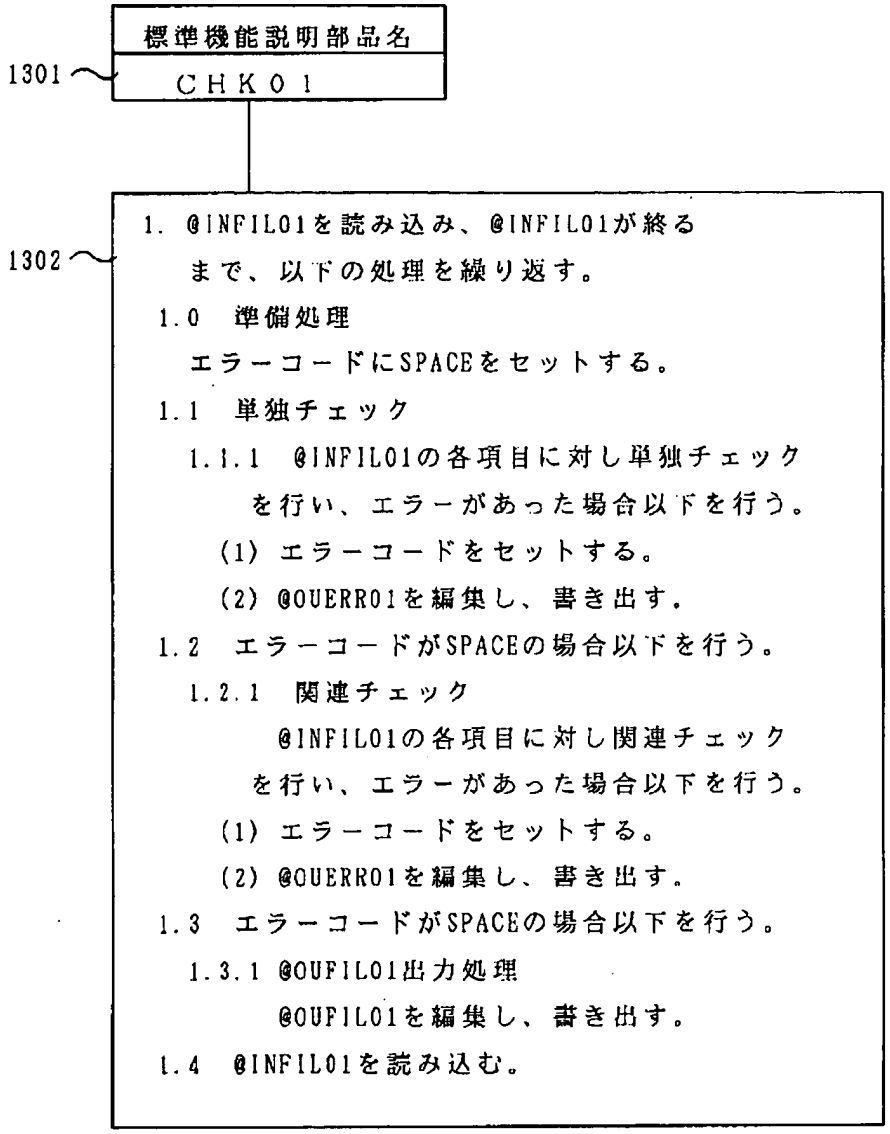
[Drawing 16]

標準パターン部品　　図16



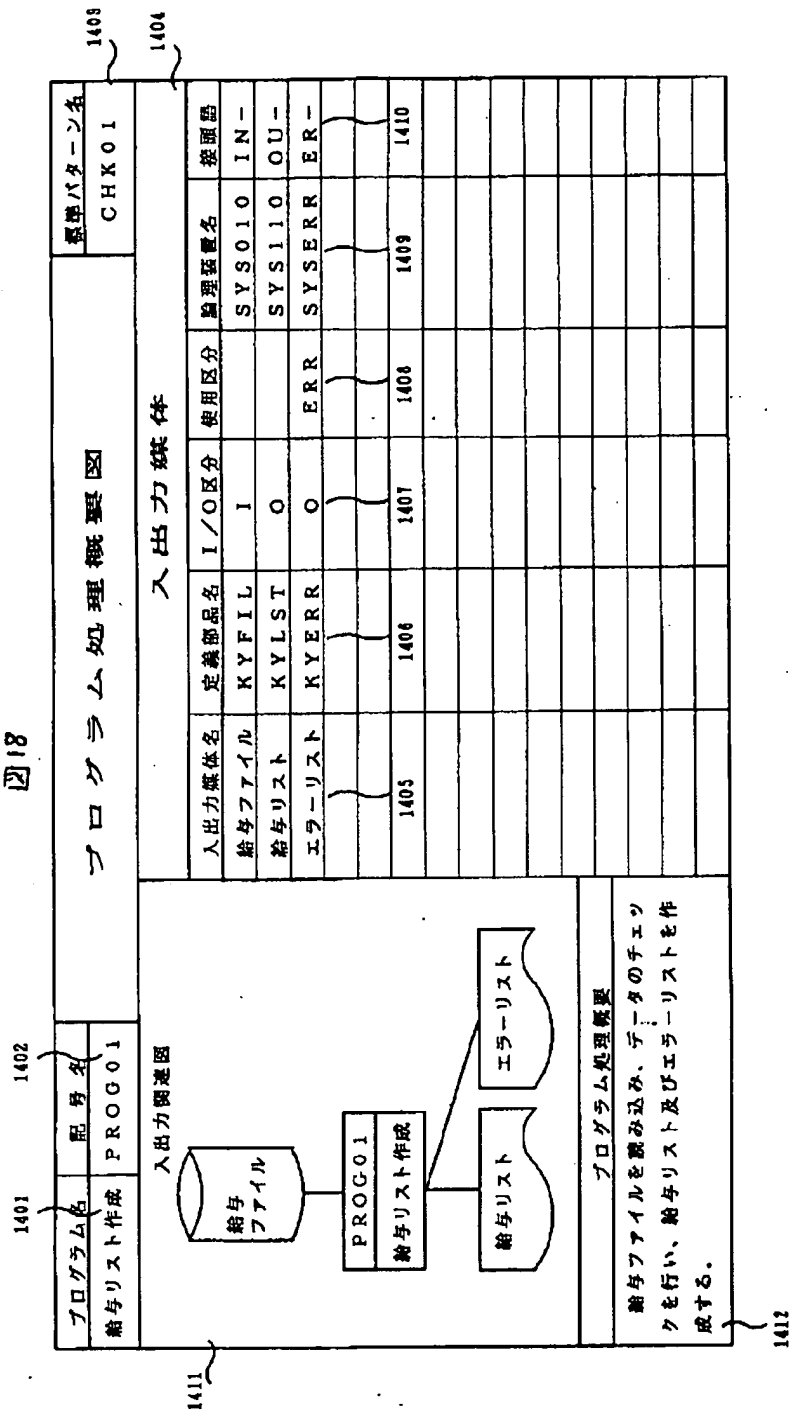
[Drawing 17]

標準機能説明部品　　図17



[Drawing 18]

図18



[Drawing 19]

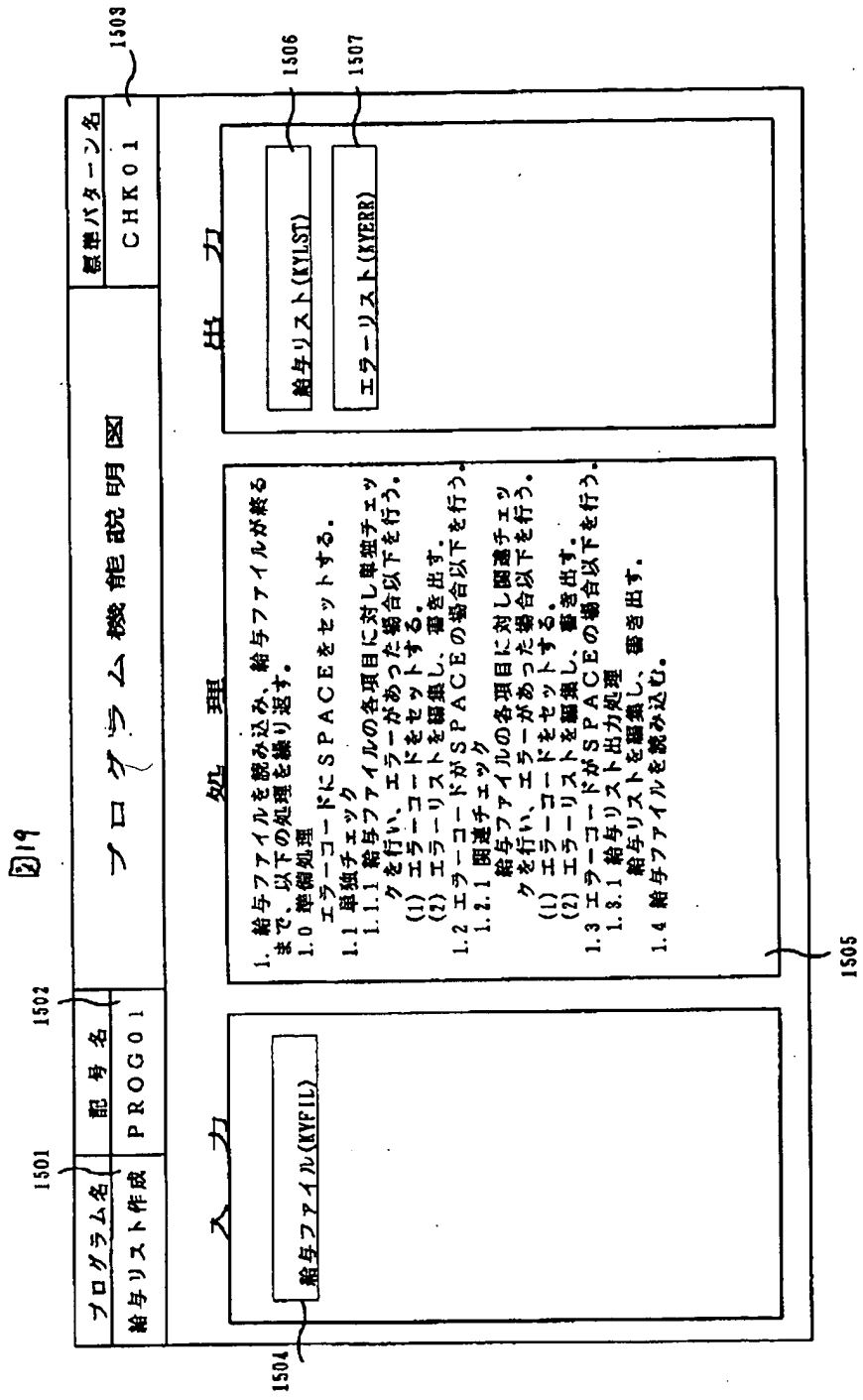


図19
プログラム機能説明図
プログラム名
給与リスト作成
記号名
PROG01
基準パターン名
CHK01
1501
1502
1503
入力
給与ファイル(KYFIL)
処理
1. 給与ファイルを読み込み、給与ファイルが終るまで、以下の処理を繰り返す。
1.0 準備処理
エラーコードにSPACEをセットする。
1.1 単独チェック
1.1.1 給与ファイルの各項目に対し単独チェックを行い、エラーがあった場合以下を行う。
(1) エラーコードをセットする。
(2) エラーリストを編集し、書き出す。
1.2 エラーコードがSPACEの場合以下を行う。
1.2.1 関連チェック
給与ファイルの各項目に対し関連チェックを行い、エラーがあった場合以下を行う。
(1) エラーコードをセットする。
(2) エラーリストを編集し、書き出す。
1.3 エラーコードがSPACEの場合以下を行う。
1.3.1 給与リスト出力処理
給与リストを編集し、書き出す。
1.4 給与ファイルを読み込む。
出力
給与リスト(KYLST)
エラーリスト(KYERR)
1504
1505
1506
1507

[Drawing 20]

図20

入力元	定義部品名	チェック条件表	プログラム名	記号名	出力先	定義部品名
給与ファイル	KYFIL		給与リスト作成	PROG01	エラーリスト	KYERR

入力項目		チェック処理			
入力項目名	属性	ｷｰﾜｰﾄﾞ	チェック条件	ｴﾗｰｺｰﾄﾞ	エラーメッセージ
氏名番号 SIMEI-NO	9(4)	SUB(SUB01)	サブルーチン(SUB01)を呼び出すことによりチェックする。	ER10	'氏名番号が正しくありません'
入社年月日 NYUSYA-YMD	9(6)	SUB(SUB03)	サブルーチン(SUB03)を呼び出すことによりチェックする。	ER20	'入社年月日が正しくありません'
職位コード SYOKUI-CD	X(3)	SEARCH (SYKTBL)	テーブル(SYKTBL)をサーチしてチェックする。	ER30	'職位コードが正しくありません'
支給額 SIKYU-GK	9(7)	RANGE(>=0)	0以上であること。	ER40	'支給額が0以上ではありません。

1601 1602 1603 1604 1605 1606 1607 1608 1609 1610 1611 1612

[Drawing 32]

図32

ディクショナリ（逆生成後）

データ項目名	NYUSYA-YMD
日本語名	入社年月日
属性	9(6)
コメント	従業員の入社した日 西暦

チェック	
条　件	SUB(SUB03)
ｴﾗｰｺｰﾄﾞ	'ER20'
ｴﾗｰ ﾒｯｾｰｼﾞ	'入社年月日が 正しくありません'

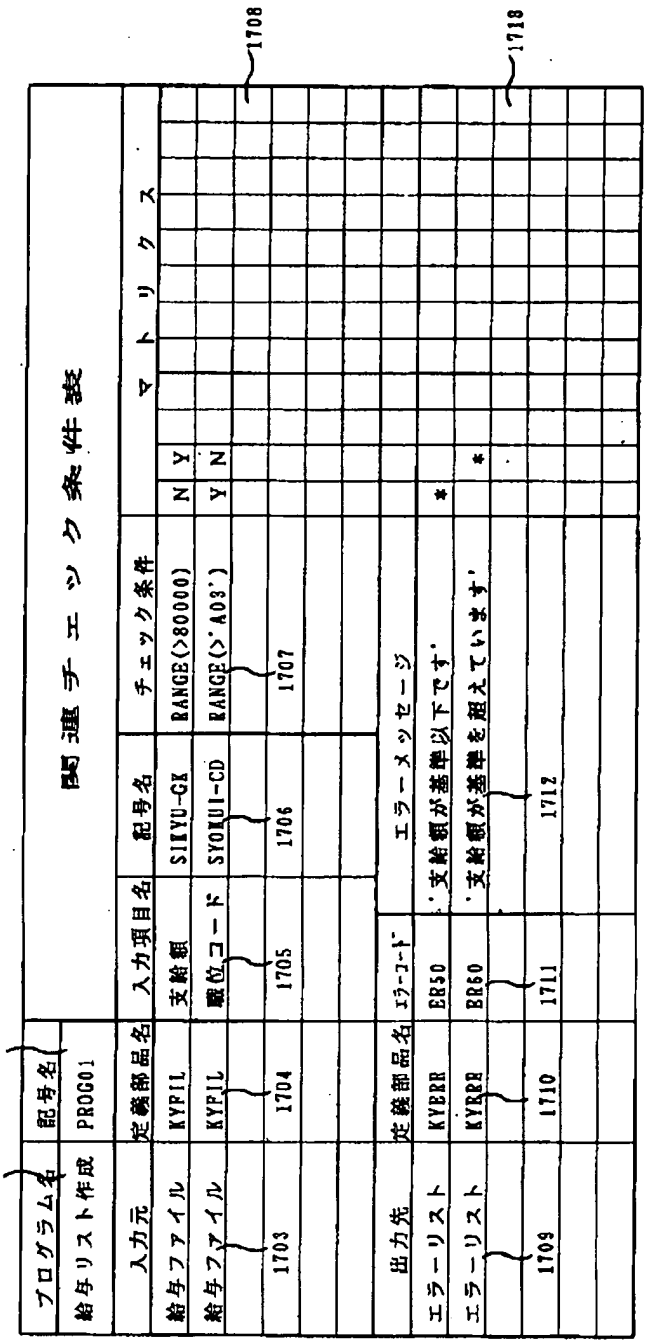
入力編集

出力編集

2401

[Drawing 21]

図21



[Drawing 24]

図24

(a)

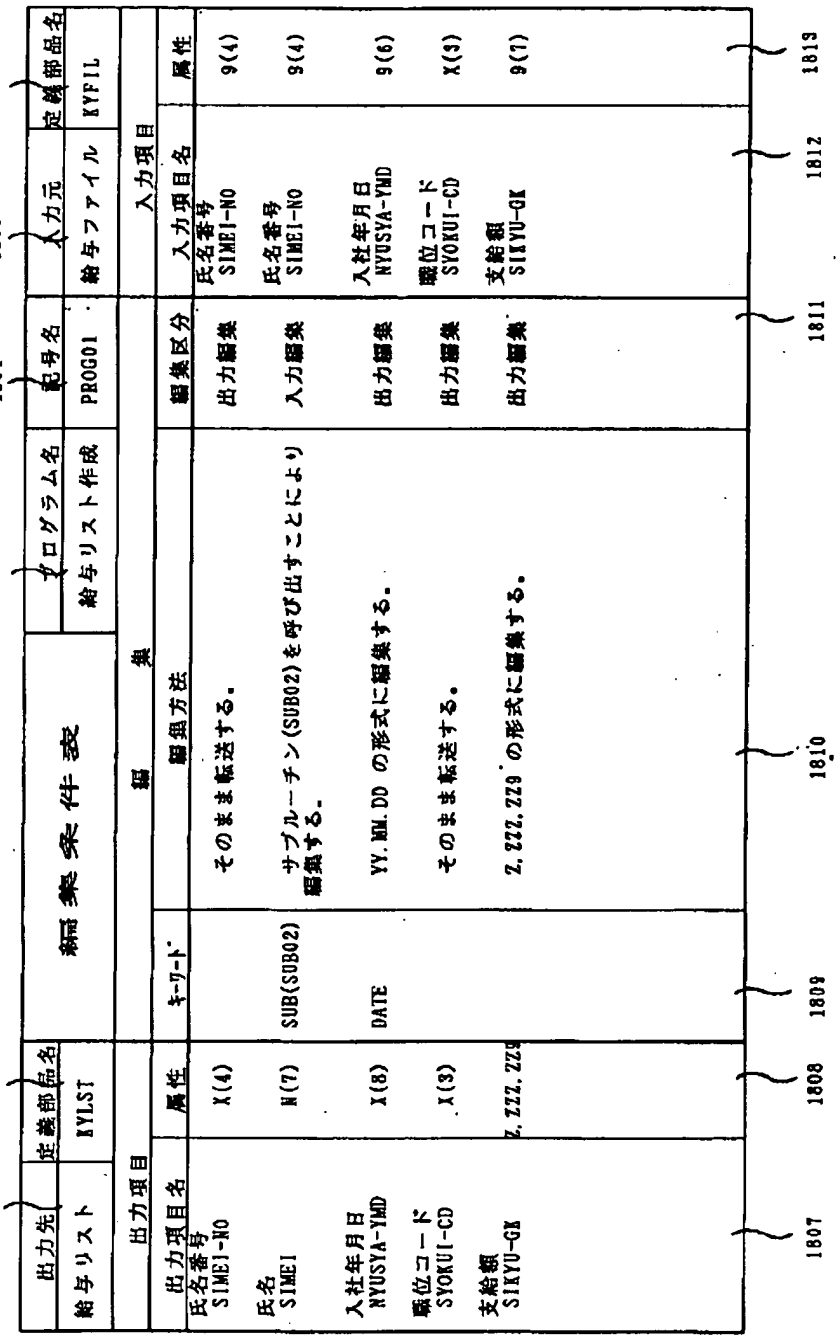
2000～項番	入力元	出力先	入力項目名	配号名	属性	キーコード	チェック条件	エラーコード	エラーメッセージ
1	KYFIL	KYERR	氏名番号	SIMEI-NO	9(4)	SUB(SUB01)	サブルーチン(SUB01)を呼び出すことによ…	ER10	・氏名番号が正しくありません。
2	KYFIL	KYERR	入社年月日	NYUSYA-YMD	9(6)	SUB(SUB03)	サブルーチン(SUB03)を呼び出すことによ…	ER20	・入社年月日が正しくありません。
3	KYFIL	KYERR	職位コード	SYOKUI-CD	X(3)	SEARCH(SYKTBL)	テーブル(SYKTBL)をサーチしてチェック…	ER30	・職位コードが正しくありません。
4	KYFIL	KYERR	支給額	SIKYU-GK	9(7)	RANGE(>=0)	0以上であること。	ER40	・支給額が0以上ではありません。

(b) 修正後

項番	入力元	出力先	入力項目名	配号名	属性	キーコード	チェック条件	エラーコード	エラーメッセージ
1	KYFIL	KYERR	氏名番号	SIMEI-NO	9(4)	SUB(SUB01)	サブルーチン(SUB01)を呼び出すことによ…	ER10	・氏名番号が正しくありません。
2	KYFIL	KYERR	入社年月日	NYUSYA-YMD	9(6)	SUB(SUB03)	サブルーチン(SUB03)を呼び出すことによ…	ER20	・入社年月日が正しくありません。
3	KYFIL	KYERR	職位コード	SYOKUI-CD	X(3)		チェックなし。		
4	KYFIL	KYERR	支給額	SIKYU-GK	9(7)	NUMERIC	数字であること。	ER44	・支給額が数字ではありません。

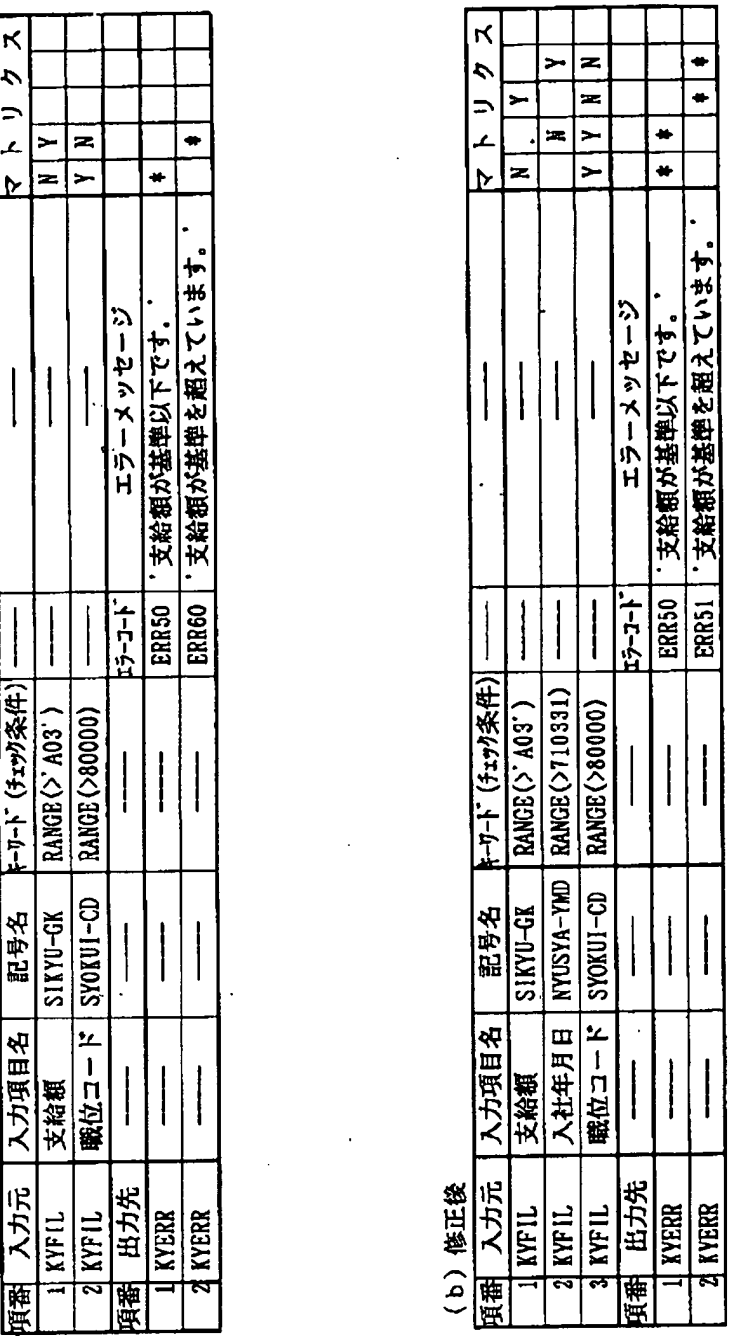
[Drawing 22]

図22



[Drawing 25]

図25



(a)

2100～

項番	入力元	入力項目名	記号名	キーワード(チェック条件)			マトリクス		
	2101	2102	2103	2104	2105	2106			2107
1	KYF1L	支給額	SIKYU-GK	RANGE(>'A03')	——	——	N	Y	
2	KYF1L	職位コード	SYOKUI-CD	RANGE(>80000)	——	——	Y	N	

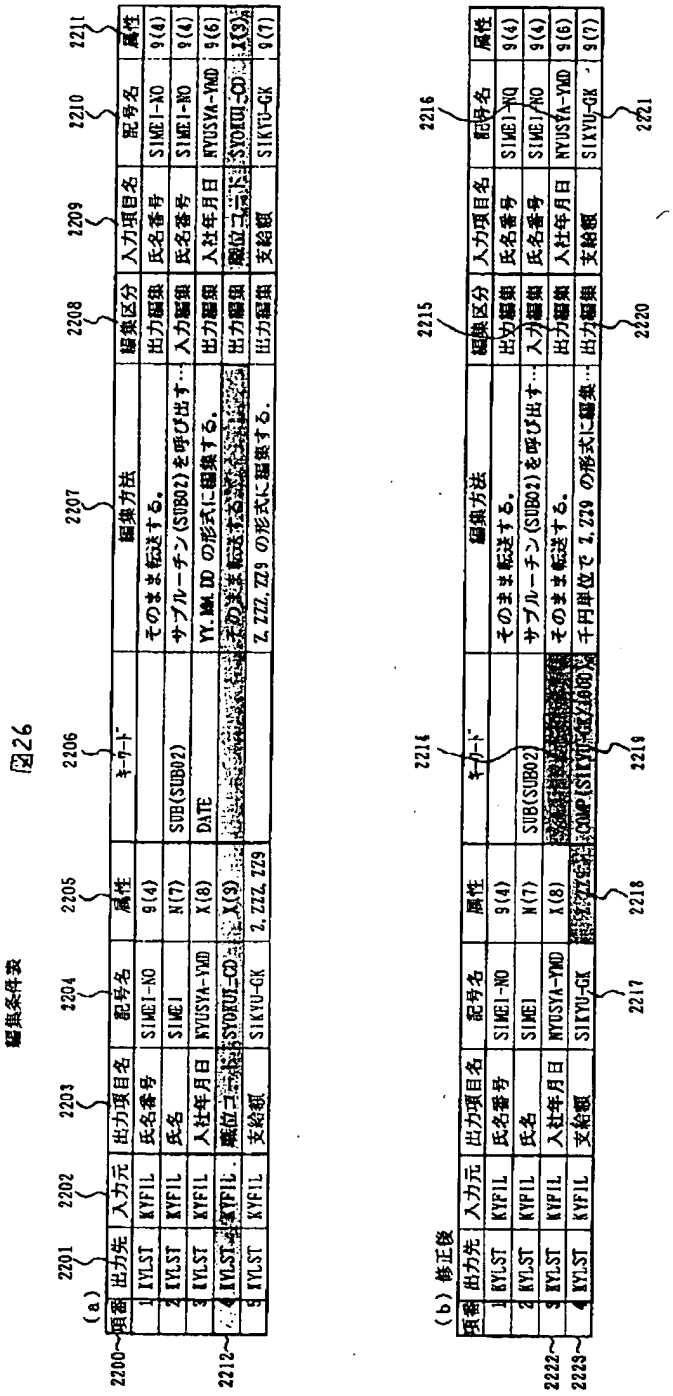
項番	出力先	——	——	——	エラーコード	エラーメッセージ			
1	KYERR	——	——	——	ERR50	'支給額が基準以下です。'	*		
2	KYERR	——	——	——	ERR60	'支給額が基準を超えています。'		*	

(b) 修正後

項番	入力元	入力項目名	記号名	キーワード(チェック条件)	——	——	マトリクス		
1	KYF1L	支給額	SIKYU-GK	RANGE(>'A03')	——	——	N	Y	
2	KYF1L	入社年月日	NYUSYA-YMD	RANGE(>710331)	——	——		N	Y
3	KYF1L	職位コード	SYOKUI-CD	RANGE(>80000)	——	——	Y	N	N

項番	出力先	——	——	——	エラーコード	エラーメッセージ			
1	KYERR	——	——	——	ERR50	'支給額が基準以下です。'	*	*	
2	KYERR	——	——	——	ERR51	'支給額が基準を超えています。'		*	*

[Drawing 26]



編集条件表

図26

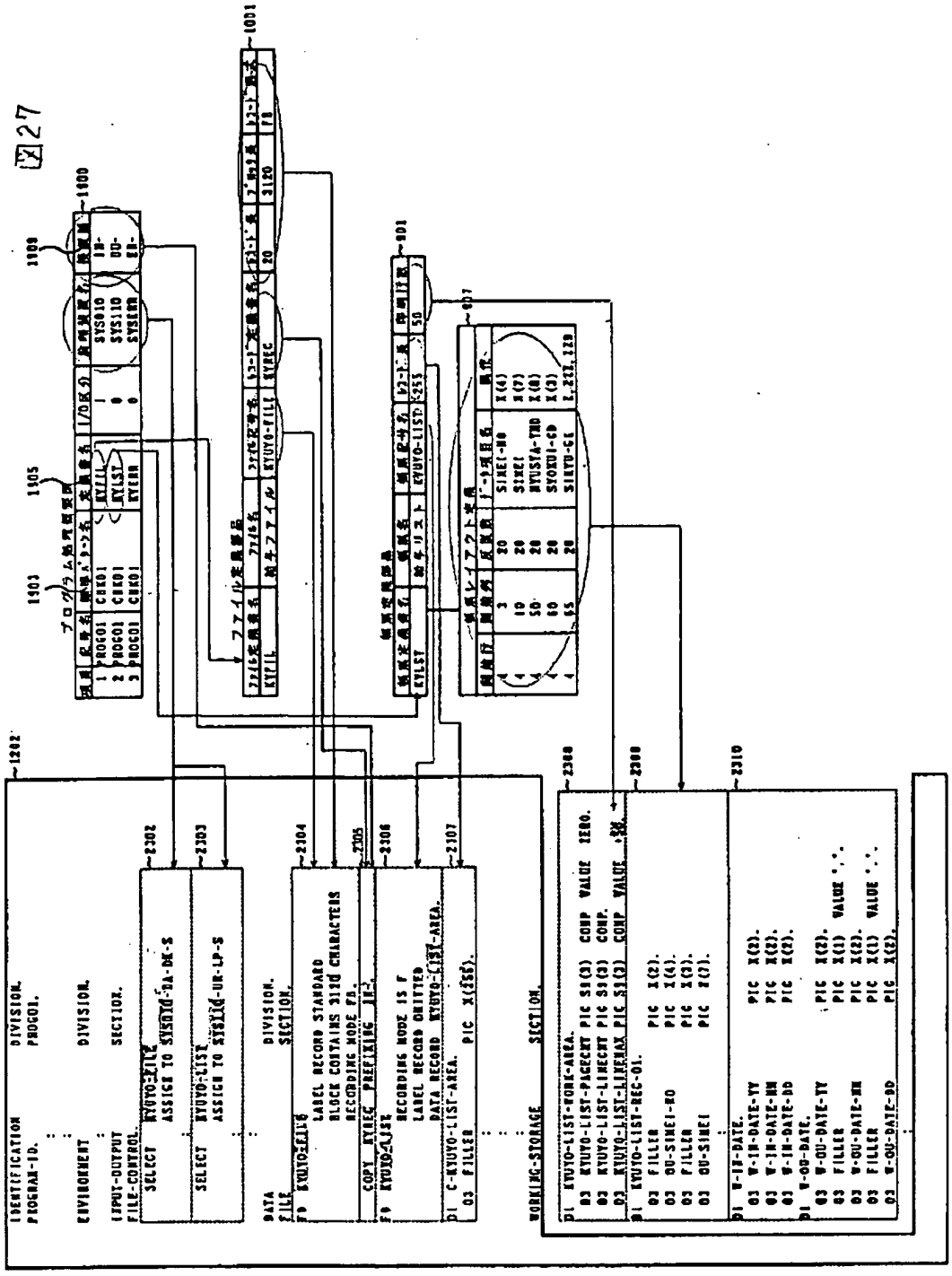
(a)

順番	出力先	入力元	出力項目名	記号名	属性	キーワード	編集方法	編集区分	入力項目名	記号名	属性
1	KYLST	KYFIL	氏名番号	SIMEI-NO	9(4)		そのまま転送する。	出力編集	氏名番号	SIMEI-NO	9(4)
2	KYLST	KYFIL	氏名	SIMEI	N(7)	SUB(SUB02)	サブルーチン(SUB02)を呼び出す…	入力編集	氏名番号	SIMEI-NO	9(4)
3	KYLST	KYFIL	入社年月日	NYUSYA-YMD	X(8)	DATE	YY.MM.DD の形式に編集する。	出力編集	入社年月日	NYUSYA-YMD	9(6)
4	KYLST	KYFIL	職位コード	SYOKUI_CD	X(3)		そのまま転送する。	出力編集	職位コード	SYOKUI_CD	X(3)
5	KYLST	KYFIL	支給額	SIKYU-GK	Z,ZZZ,ZZ9		Z,ZZZ,ZZ9 の形式に編集する。	出力編集	支給額	SIKYU-GK	9(7)

(b) 修正後

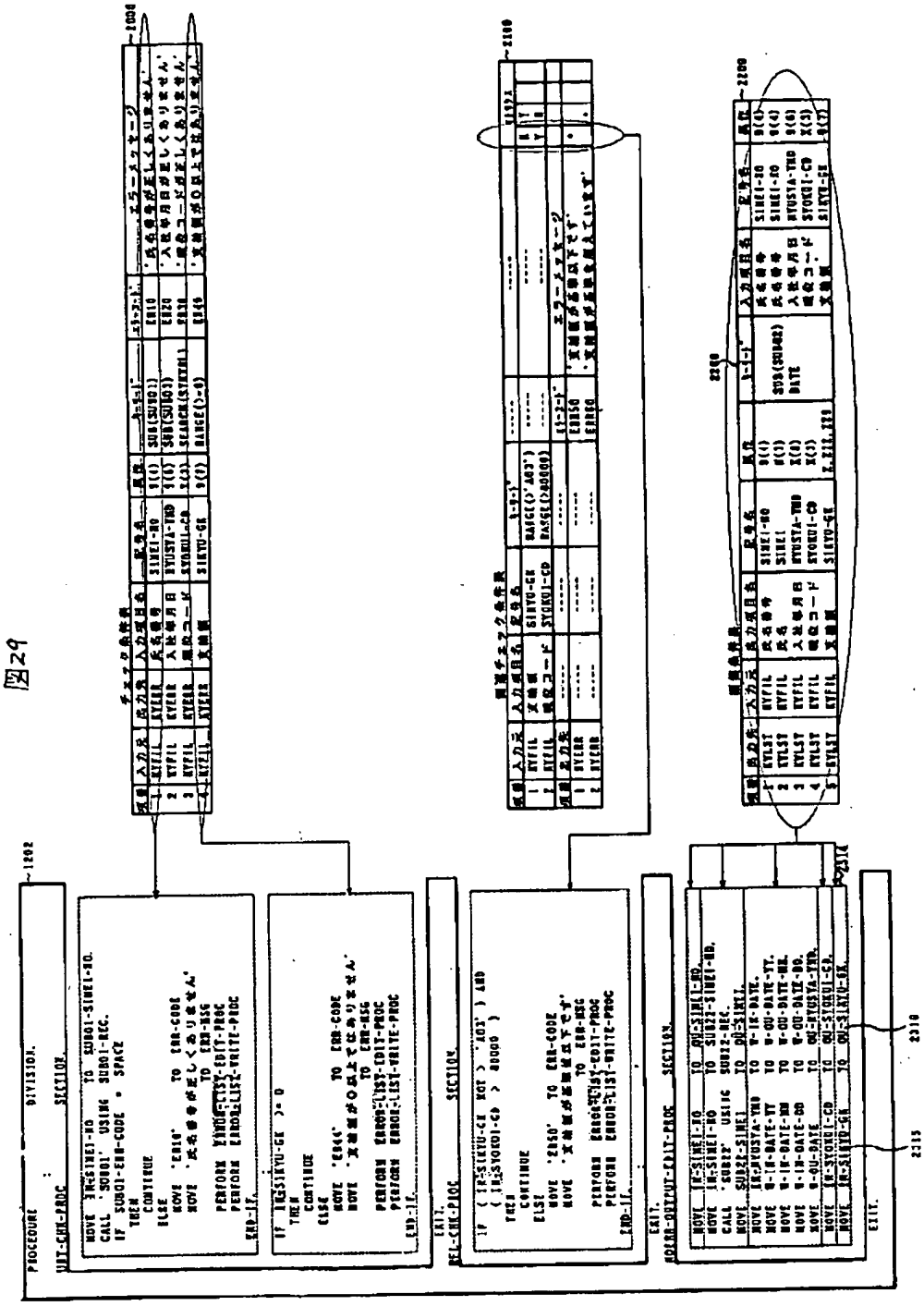
順番	出力先	入力元	出力項目名	記号名	属性	キーワード	編集方法	編集区分	入力項目名	記号名	属性
1	KYLST	KYFIL	氏名番号	SIMEI-NO	9(4)		そのまま転送する。	出力編集	氏名番号	SIMEI-NO	9(4)
2	KYLST	KYFIL	氏名	SIMEI	N(7)	SUB(SUB02)	サブルーチン(SUB02)を呼び出す…	入力編集	氏名番号	SIMEI-NO	9(4)
3	KYLST	KYFIL	入社年月日	NYUSYA-YMD	X(8)	[redacted]	そのまま転送する。	出力編集	入社年月日	NYUSYA-YMD	9(6)
4	KYLST	KYFIL	支給額	SIKYU-GK	[redacted]	COMP(SIKYU-GK,2)	千円単位で Z,ZZ9 の形式に編集…	出力編集	支給額	SIKYU-GK	9(7)

[Drawing 27]



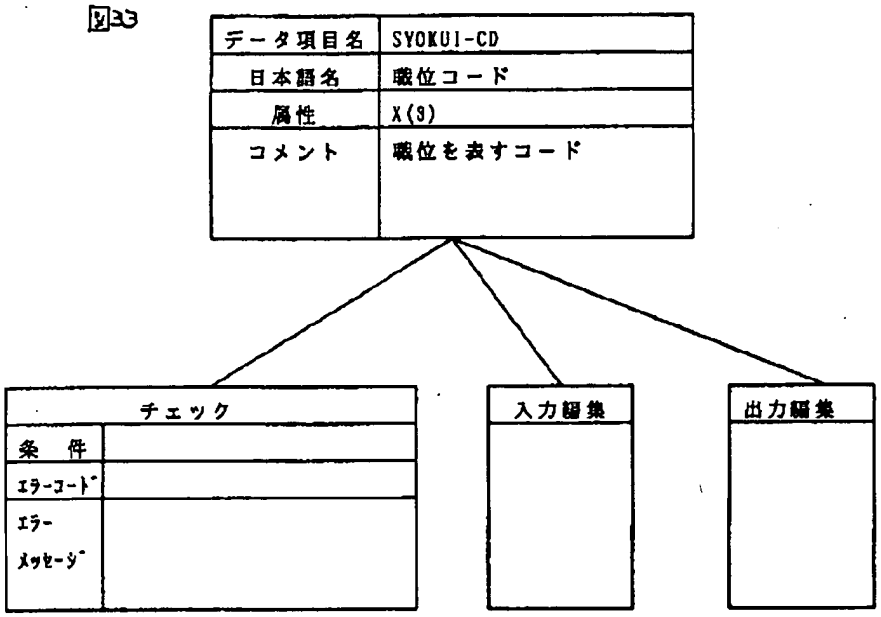
[Drawing 29]

図29



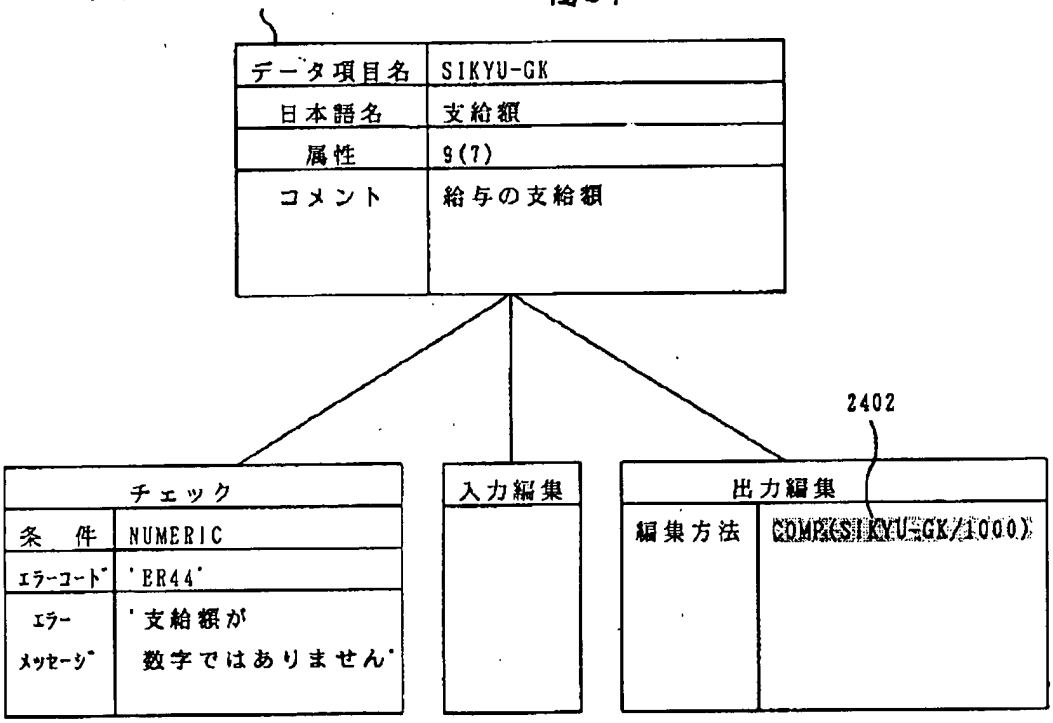
[Drawing 33]

図33



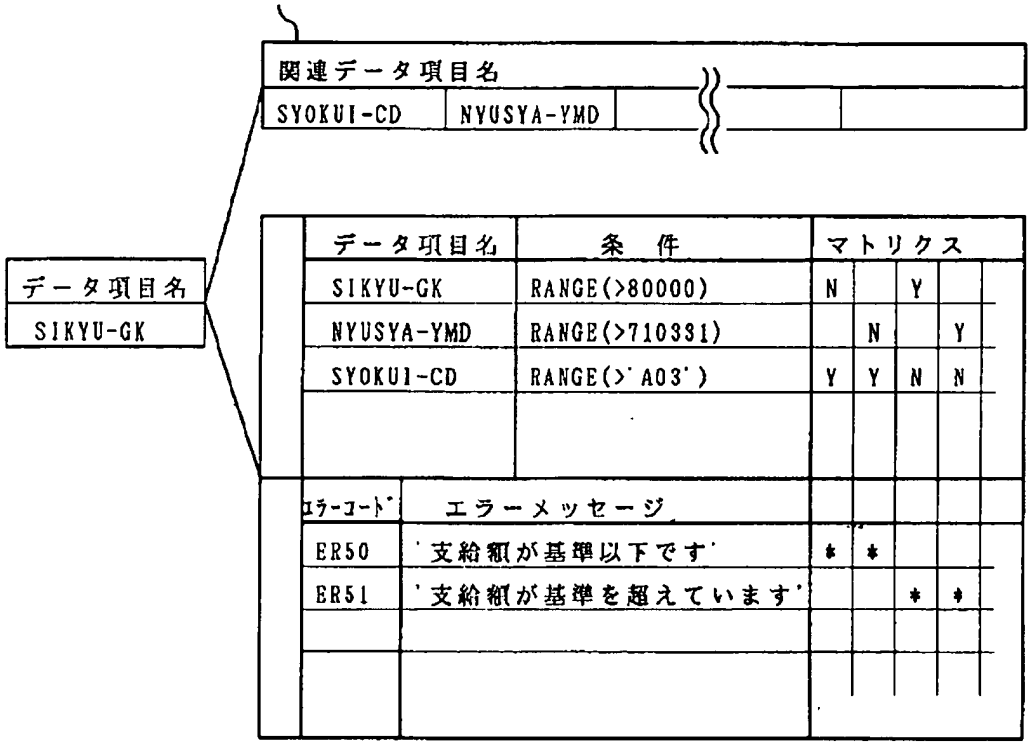
[Drawing 34]

ディクショナリ（逆生成後）　図34



[Drawing 35]

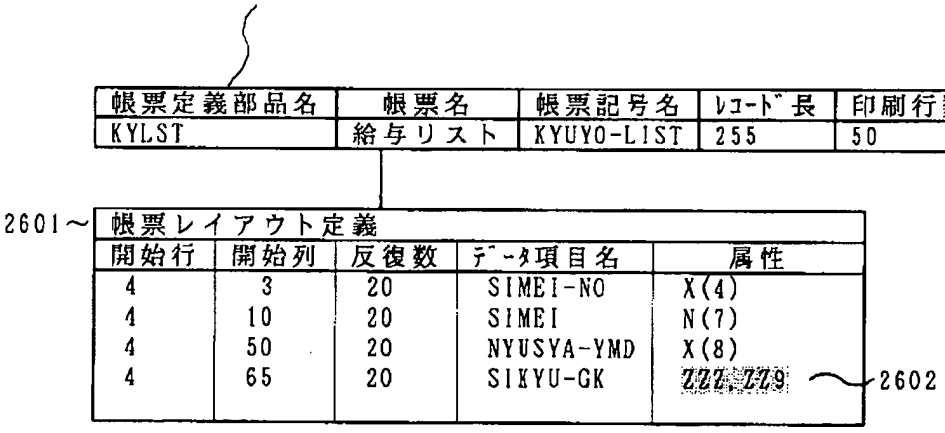
関連チェック定義部品　　図35



[Drawing 36]

図36

帳票定義部品（逆生成後）



[Translation done.]